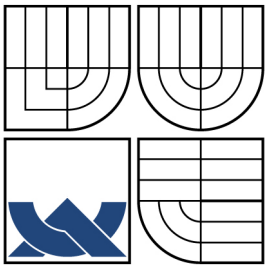
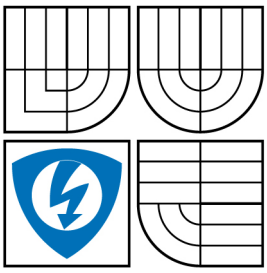


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

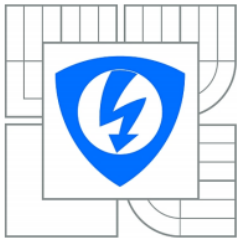
SBĚR DAT O SÍŤOVÉ KOMUNIKACI ZE ZAŘÍZENÍ SÍŤOVÉ INFRASTRUKTURY

DIPLOMOVÁ PRÁCE

AUTOR PRÁCE
AUTHOR
LUKÁŠ GARGULÁK

VEDOUCÍ PRÁCE
SUPERVISOR
ING. RADKO KRKOŠ

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Lukáš Gargulák	***ID:***	106444
Ročník:	2	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Sběr dat o síťové komunikaci ze zařízení síťové infrastruktury

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte problematiku získávání statistických dat o síťové komunikaci ze zařízení síťové infrastruktury v sítích s komutací datových jednotek. Při studiu se zaměřte zejména na směrovače a přepínače a na možnosti protokolu SNMP při vzdáleném získávání statistických dat. Ověřte i další možné postupy vzdáleného získávání dat. Různé nastudované metody popište a komentujte jejich výhody a nevýhody. Analyzované postupy vybírejte s ohledem na jednoduchost nasazení, minimální množství zásahů do sledované datové sítě a minimalizaci vlivu měření na parametry sítě. Důležitými parametry jsou detailnost a obsáhlost získaných dat. Navrhněte systém, který umožní získat statistické údaje ze zařízení síťové infrastruktury a tyto údaje uloží ve vhodném formátu, k dispozici nechť je také možnost exportu všech dat pro externí zpracování. Navržený systém umožní získané údaje pomocí webové aplikace zobrazovat ve formě grafů a tabulek. Tato aplikace bude schopná operátorovi podat představu o topologii sítě. Vytvořte laboratorní úlohu demonstrující získané znalosti.

DOPORUČENÁ LITERATURA:

[1] MAURO, Douglas; SCHMIDT, Kevin. Essential SNMP. Second Edition. [s.l.] : O'Reilly Media, 2005. 464 s. ISBN 978-0-596-00840-6.
[2] SHIPWAY, Steve. Using MRTG with RRDtool and Routers2. Third Edition. [s.l.] : Cheshire Cat Computing, 2010. 357 s.

Termín zadání:	6.2.2012	***Termín odevzdání:***	24.5.2012

Vedoucí práce: Ing. Radko Krkoš
Konzultanti diplomové práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

Abstrakt

V teoretické části diplomové práce jsou srovnávány protokoly vhodné pro sběr statistických dat ze zařízení síťové infrastruktury. Vhodným protokolem pro vytvoření aplikace k tomuto účelu byl zvolen protokol SNMP, jelikož je nejrozšířenější. Práce se zabývá detailní teorií okolo protokolu SNMP včetně popisu MIB databáze a seznámení se s SNMP operacemi. Dále je vytvořena aplikace pro vytvoření topologie sítě, která správci vytvoří představu nad dohledovanou síti. Pro každé nalezené zařízení s podporou SNMP protokolu jsou periodicky sbírána a ukládána statistická data jež je možné exportovat do souboru. Diplomová práce obsahuje také dvě přílohy. Jedna obsahuje zadání laboratorní úlohy pro studenty a druhá příloha v podobě DVD obsahuje hotovou aplikaci SDSKSI - Sběr dat o síťové komunikaci ze zařízení síťové infrastruktury.

Abstract

The diploma thesis describes theory that is needed for application development for acquisition of communication statistical data from network infrastructure devices. Aplication is called SDSKSI. The project compares protocols suitable for this purpose. Finally SNMP protocol was chosen because it is the most common in network devices. SNMP is described in detail. Each SNMP operation has its own practical demonstration. In the project there is also described MIB database and data types of MIB objects. Application is able to create network topology. Then administrator of network can imagine how the network looks like. For each device that support SNMP protocol are periodically collected and stored statistical data which can be exported to the file. For application development were chosen programming languages according to several criteria. Content of the laboratory exercise is presented. At the end of the project there are some system solutions for collecting statistical data. Diploma thesis contents two attachments. The first is containing the full text of laboratory task. The second is DVD disc. Disc is containing ready to boot aplication SDSKSI.

Klíčová slova

SNMP, NetFlow, MIB, topologie sítě, sběr dat, směrovač, přepínač, Debian

Keywords

SNMP, NetFlow, MIB, network topology, collecting data, router, switch, Debian

GARGULÁK, L. *Sběr dat o síťové komunikaci ze zařízení síťové infrastruktury*: Diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 94 stran. Vedoucí práce Ing. Radko Krkoš.

Prohlášení

Prohlašuji, že svou diplomovou práci na téma Sběr dat o síťové komunikaci ze zařízení síťové infrastruktury jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ..... …....

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Radku Krkošovi za účinnou metodickou a odbornou pomoc a další cenné rady při zpracování této práce.

Rád bych na tomto místě poděkoval Miroslavu Kamenářovi za poskytnutí síťového hardware a umožnění testování vytvářené aplikace v reálné počítačové síti. Dále bych rád poděkoval Ing. Rostislavu Jendřejčíkovi za poskytnutí rad a zkušeností s protokolem SNMP.

Za podporu při tvorbě této diplomové práce děkuji své rodině a Ivaně Žákové.

V Brně dne ....

podpis autora

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication

Brno University of Technology
Purkynova 118, CZ-61200 Brno, Czechia

http://www.six.feec.vutbr.cz

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

Obsah

Úvod....1

1 Možnosti vzdáleného získávání statistických dat o síťové komunikaci ze síťové infrastruktury....3

1.1 Simple Network Management Protocol....3

1.1.1 Historie protokolu SNMP a jeho verze....3

1.1.2 Cíle a princip SNMP....4

1.1.3 Bezpečnost SNMP a přístupová práva SNMP managerů k SNMP agentům....5

1.2 NetFlow protokol....6

1.2.1 NetFlow architektura....6

1.2.2 NetFlow paket....8

1.3 Další protokoly pro sledování komunikace v síti....9

1.4 Srovnání výhod a nevýhod protokolu SNMP a NetFlow z hlediska jednoduchosti nasazení na sledovanou síť....9

1.4.1 Výhody protokolu SNMP....9

1.4.2 Nevýhody protokolu SNMP....10

1.4.3 Výhody protokolu NetFlow....10

1.4.4 Nevýhody protokolu NetFlow....10

1.4.5 Výběr protokolu pro diplomovou práci....10

2 Komunikační protokol SNMP....11

2.1 Formát SNMP paketu....12

3 MIB databáze....14

3.1 Přístup k objektu v MIB databázi....14

3.2 Rozdělení MIB....15

3.3 Notace SMI....16

3.4 Popis notace SMIv1....16

3.4.1 Identifikátor objektu - OID....16

3.4.2 Datové typy objektů....17

3.5 Rozšíření v notaci SMIv2....18

3.6 Ukázka a popis MIB souboru....21

4 Popis operací protokolu SNMP....27

4.1 Operace get....27

4.2 Operace get-next....28

4.3 Operace get-bulk....30

4.4 Operace set....31

4.5 Operace trap....32
4.6 Operace notification....33
4.7 Operace inform....34
4.8 Operace report....34
4.9 Hlášení o chybě pro operace get, get-next, get-bulk a set....34
5 Hotová řešení pro sběr dat o síťové komunikaci....36
5.1 MRTG....36
5.2 CACTI....36
5.3 NAGIOS....36
5.4 Komerční aplikace pro sledování sítě....37
6 Popis aplikace - praktická část....38
6.1 Použité programovací jazyky....38
6.2 Vytvoření databáze pro uložení statistických dat....38
6.3 Vytvoření topologie sítě....39
6.3.1 Vstupní rozsah....40
6.3.2 První úroveň....40
6.3.3 N-tá úroveň....41
6.3.4 Generování vzhledu topologie....42
6.4 Aktuální statistická data ze síťového zařízení....43
6.5 Sběr statistik o síťovém provozu....47
6.5.1 Průměrování statistik....47
6.5.2 Vytvoření grafu propustnosti....48
6.5.3 Promazávání starých statistik....51
6.5.4 Cron....51
6.6 Export statistických dat....52
7 Instalace potřebného software a aplikace SDSKSI....53
7.1 SSH....53
7.2 Server Apache a PHP....53
7.3 Balíčky pro podporu protokolu SNMP....54
7.4 Databáze MySQL....54
7.5 GraphViz....54
7.6 Cron Job....55
7.7 Práva a vlastnictví adresářů a souborů....55
8 Popis webového rozhraní aplikace....56
9 Laboratorní úloha....60

10 Závěr.....61
Literatura.....63
Seznam zkratek.....65
A Laboratorní úloha.....67
B Obsah přiloženého DVD disku.....84

Seznam obrázků

Obr. 1.1: Princip SNMP....4
Obr. 1.2: Tradiční architektura pro NetFlow....7
Obr. 1.3: Moderní architektura pro NetFlow....8
Obr. 2.1: TCP/IP komunikační model....11
Obr. 2.2: SNMP zpráva zapouzdřená do IP paketu....12
Obr. 2.3: SNMP paket....12
Obr. 3.1: Část struktury MIB databáze dle RFC 1213 14
Obr. 3.2: Rozšířený strom objektů ve verzi SMIv2....18
Obr. 4.1: Princip SNMP operace get....27
Obr. 4.2: Hledání podstromu system pomocí příkazu snmpwalk....29
Obr. 4.3: Princip SNMP operace get-bulk....30
Obr. 4.4: Princip SNMP operace set....31
Obr. 4.5: Princip SNMP operace trap....32
Obr. 6.1: Struktura tabulek v databázi MySQL....39
Obr. 6.2: Struktura diagramu....40
Obr. 6.3: Vygenerovaná topologie sítě programem GraphViz....43
Obr. 6.4: Směrovací tabulka z webového rozhraní aplikace SDSKSI....46
Obr. 6.5: Graf propustnosti na rozhraní směrovače vygenerovaný pomocí JpGraph....50
Obr. 8.1: Snímek vzhledu aplikace po přihlášení uživatele s menu na levé straně....56
Obr. 8.2: Nastavení podsítě první úrovně....56
Obr. 8.3: Vytvoření topologie sítě....57
Obr. 8.4: Topologie sítě....57
Obr. 8.5: Statistická data o konkretním zařízení v síti s podporou SNMP protokolu....58
Obr. 8.6: Graf propustnosti za posledních 60 minut směrovače s IP adresou 10.101.0.10....59

Úvod

Tato diplomová práce se zabývá vytvořením aplikace, která umožní vzdáleně získávat statistická data o síťové komunikaci ze zařízení síťové infrastruktury a zároveň dokáže správci sítě podat představu o topologii sítě. Aplikace je pojmenována zkratkou SDSKSI – **S**běr **D**at o **S**íťové **K**omunikaci ze zařízení **S**íťové **I**nfrastruktury.

V první kapitole jsou rozebírány možnosti vzdáleného získávání statistických dat o síťové komunikaci ze síťové infrastruktury. Je rozebrán protokol SNMP (Simple Network Management Protocol) a protokol NetFlow od firmy Cisco Systems. Jsou zde také zmíněny některé další protokoly pro získávání statistických dat. U protokolu SNMP je rozebrána také problematika bezpečnosti. V závěru kapitoly jsou shrnuty výhody a nevýhody uvedených protokolů z hlediska jednoduchosti nasazení na sledovanou síť.

V další kapitole je více do hloubky rozebrán protokol SNMP a je popisován formát SNMP paketu přenášeného sítí.

Třetí kapitola pojednává o MIB (Management Information Base) databázi. Je možné se dozvědět, co to MIB databáze je, k čemu se používá a jakou má strukturu. Je zde také první zmínka o tom, jak se přistupuje k tzv. objektu v MIB databázi. V kapitole je popsána notace SMI (Structure of Managed Information) první i druhé verze. Jsou zde uvedeny všechny datové typy objektů a taktéž je podrobněji popsána problematika tzv. identifikátoru objektu – OID (Object Identifier). Na konci kapitoly je uvedena ukázka MIB souboru. Jednotlivé řádky souboru jsou postupně a podrobně vysvětleny.

Čtvrtá kapitola popisuje jednotlivé operace protokolu SNMP a pro většinu operací jsou provedeny praktické ukázky, kde je vypsán výpis z konzole.

Na konci teoretické části diplomové práce jsou v páté kapitole uvedena hotová řešení pro sběr statistických dat o síťové komunikaci a jsou popsány jejich výhody a nevýhody.

Praktická část je rozebírána počátkem šesté kapitoly. V této kapitole jsou ze začátku uvedeny používané programovací jazyky a následně struktura MySQL databáze. Dále jsou podrobně rozebírány skripty, které zajišťují sestavení topologie sítě a sběr statistických dat o síťové komunikaci ze zařízení síťové infrastruktury. Konec kapitoly je věnován problematice exportu statistických dat pro další zpracování.

Sedmá kapitola pojednává o instalaci všech potřebných balíčku do operačního systému Debian GNU/Linux, na němž je celá aplikace SDSKSI postavena. Důležitá nastavení pro správný běh aplikace jsou podrobně popsána.

Samotné webové rozhraní aplikace SDSKSI, ke kterému má správce sítě přístup, je popsáno v kapitole osmé.

Poslední kapitola se krátce věnuje obsahu laboratorní úlohy pro studenty UTKO FEKT VUT Brno. Kompletní laboratorní úloha je doložena v příloze A. Její potřebné softwarové vybavení je přiloženo na disku DVD.

1 Možnosti vzdáleného získávání statistických dat o síťové komunikaci ze síťové infrastruktury

V následujícím textu budou uvedeny dvě možnosti pro vzdálené získávání statistických dat o síťové komunikaci ze síťové infrastruktury.

1.1 Simple Network Management Protocol

Pro sběr statistických dat o síťové komunikaci je možné využít Simple Network Management Protocol, dále jen SNMP. Tyto data získává protokol SNMP ze sledovaného prvku pomocí SNMP agenta. Před samotným principem SNMP bude stručně popsána historie tohoto protokolu [3].

1.1.1 Historie protokolu SNMP a jeho verze

Protokol SNMP se vyvinul z protokolu SGMP (Simple Gateway Monitoring Protokol). Začal se využívat v roce 1988 a brzy se stal nejrozšířenějším protokolem pro řízení a sledování počítačových sítí. Podpora SNMP protokolu je v dnešní době rozšířena nejen u serverů a aktivních síťových prvků, ale téměř u všech zařízení, které komunikují po počítačové síti (tiskárny, záložní zdroje, bezdrátové přístupové body, či různá síťová čidla, jako například teploměr apod.).

SNMPv1

První verze tohoto protokolu, SNMPv1, je specifikována v RFC[1] 1157. Obsahuje řadu nedostatků. Například autentizace pomocí hesla, tzv. *community string*, je nedostatečná. Toto heslo se přes sít nepřenáší v šifrované formě. Bezpečnosti při přenosu dat přes protokol SNMP se věnuje kap. 1.1.3.

SNMPv2

Verze protokolu SNMPv2 existuje v několika variantách. Tento standard začal vznikat v roce 1993 a později, v roce 1996, byl upraven. V této verzi SNMP přibyly dvě nové operace (*get-bulk* a *inform*). Tyto i jiné operace jsou popsány v kap. 1.1.3, která se věnuje operacím dostupným v SNMP. Nejrozšířenějším standardem mezi výrobci síťových zařízení je dnes standard ve variantě SNMPv2c popisovaný v RFC 1901 až RFC 1908.

1 RFC - Request for Comments

SNMPv3

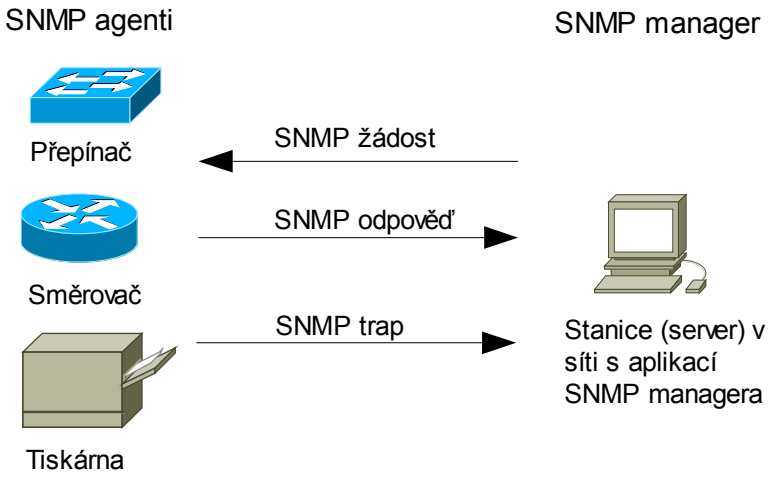
Z důvodu nedostatečné bezpečnosti obou předchozích verzí vznikla na jaře roku 1998 ještě novější verze - SNMPv3. Předchozí verze měly ochranu pro přenos dat pouze pomocí community string. V této verzi je již komunikace šifrována DES algoritmem. V současné době tento standard protokolu SNMP není tolik rozšířen mezi výrobci síťových prvků. Standard SNMPv3 je definován v RFC 3411 až RFC 3418.

1.1.2 Cíle a princip SNMP

Při vzniku protokolu SNMP byl kladen požadavek vytvořit pro administrátory sítí univerzální protokol pro sledování stavu sítě a jejich vzdálené řízení. Protokol SNMP je založený na modelu *klient* / *server*. Klient je označován jako SNMP manager a server jako SNMP agent [3].

SNMP manager

SNMP manager je program, který běží na síťové stanici. Ve větších sítích se jako manager často používá NMS (Network Management System), který běží na pracovní stanici vyhrazené pro tento účel. Pomocí NMS pak administrátor sítě může číst nebo také nastavovat jednotlivé parametry sledovaných a řízených prvků v počítačové síti pomocí dotazování SNMP agenta pomocí SNMP operací. Jak SNMP manažer komunikuje s jednotlivými agenty, je naznačeno na obr. 1.1. Manager posílá dotazy agentovi a přijímá jeho odpovědi. Hodnoty ze sledovaných prvků může získávat i více SNMP managerů najednou.



Obr. 1.1: Princip SNMP

SNMP agent

SNMP agent je program, který běží na zařízení připojeném do počítačové sítě a odpovídá na dotazy SNMP managera. Agent neustále monitoruje a sbírá informace o všech dostupných

funkcích a stavech daného zařízení. K získání určité informace ze zařízení, na kterém běží SNMP agent, musí SNMP manager vyslat požadavek na dané zařízení a projít informace poskytované agentem.

Mimo jiné SNMP agent může vysílat asynchronně hlášení o poruše (tzv. *trapy*[2]) na adresu definovanou administrátorem. Tyto trapy jsou odesílány bez žádosti SNMP managera. Může se jednat o zasílání různých hlášení o poruše zařízení v počítačové síti, výpadku napájení, výpadku ventilátorů, vysoké teplotě apod.

1.1.3 Bezpečnost SNMP a přístupová práva SNMP managerů k SNMP agentům

Přístup k jednotlivým objektům agenta je důležité nějak zabezpečit. Jedná se vlastně o tzv. přístupové práva SNMP managerů k SNMP agentům. Tohoto zabezpečení se dosáhlo velice primitivním principem. Tím, že každý paket ve své hlavičce nese pole označené *community string* [3]. Toto pole funguje podobně jako kombinace uživatelského jména a hesla.

V praxi je definován jeden *community string* pro čtení i zápis (*read-write*) a jeden *community string* pro omezený přístup, pouze pro čtení (*read-only*). Jestliže je tedy ve sledovaném zařízení uložen stejný *community string*, jako v příkazu od SNMP managera, provede se požadovaná akce podle typu operace. Nebudou-li souhlasit, bude akce odmítnuta.

Informace posílané v SNMP paketech jsou uloženy v čistém textu. To znamená, že i *community string* je přenášen jako čistý text a je možné jej síťovým analyzátorem odchytit.

Proto přišla verze SNMPv3, která přinesla několik vylepšení pro oblast bezpečnosti a komunikaci šifruje pomocí DES algoritmu.

Ve výchozím nastavení je u SNMP agentů nastaven *community string* pro čtení na hodnotu *public* a pro čtení a zápis na hodnotu *private*.

Protože *community string* je v podstatě heslo, měly by se při výběru zohledňovat obecné kriteria a pravidla pro vytvoření hesla. Neměla by se používat slovníková slova. Heslo by se mělo skládat z číslic i písmen, velkých i malých.

Je důležité si uvědomit, kdo má přístup pro čtení i zápis k SNMP zařízením a kdo tak může získat kontrolu nad těmito zařízeními použitím protokolu SNMP (např. může nastavovat rozhraní směrovače, přepínat stavy portů nebo měnit směrovací tabulky).

Jednou z cest, jak se chránit před tímto útokem je používat virtuální soukromou síť VPN (Virtual Private Network) a zajistit tak, že provoz je šifrován.

---

2 Trap - Hlášení o chybě na zařízení připojeném do sítě. Tento výraz bude používán v celém dokumentu.

Druhou z cest je častá změna *community string*. Změna *community string* není obtížná v malých sítích. Jedná-li se však o síť se stovkami síťových prvků, ze kterých jsou sbírány statistické údaje, je vhodné pro změnu *community string* napsat skript.

Pro snížení pravděpodobnosti útoku je možné nastavit IP firewall nebo filtry tak, že je v nich např. povolen UDP (User Datagram Protokol) provoz pouze od známých zařízení. Je také možné povolit UDP provoz pouze na port 161 (pro SNMP žádosti) nebo na port 162 (pro trapy). Firewall není stoprocentně efektivní řešení, ale určitě dokáže zredukovat pravděpodobnost útoku na zařízení v síti.

1.2 NetFlow protokol

NetFlow je otevřeným protokolem vyvinutým společností Cisco Systems [2], [7], [15]. Nejnovější verze 9 je definována v RFC 3954. Původně se tento standard využíval jen jako doplňková služba k Cisco směrovačům. Standard NetFlow je v posledních letech nejrozšířenějším standardem pro měření a sledování počítačové sítě na základě IP toků. Díky tomuto sledování umožňuje administrátorům nahlížet na provoz v síti.

Pomocí získaných statistik se mohou například získávat podklady pro účtování služeb na základě objemu přenesených dat. Pomocí NetFlow statistik je možné odhalit i slabá místa v síti, efektivně naplánovat rozšíření sítě, zjistit kdo je hlavním zdrojem provozu, kdo s kým na síti komunikoval, jak dlouho a pomocí kterého protokolu komunikoval apod.

NetFlow tok je určen sekvencí paketů, které obsahují pět hlavních údajů:

- zdrojovou adresu,
- cílovou adresu,
- zdrojový port,
- cílový port,
- číslo protokolu.

Pro každý NetFlow tok je také zaznamenáno časové razítko, kdy tok vznikl a jak dlouho trval.

1.2.1 NetFlow architektura

V typické NetFlow architektuře je možné naleznout několik tzv. NetFlow exportérů a jeden tzv. NetFlow kolektor. NetFlow exportér má za úlohu sledovat a analyzovat pakety na lince, ke které je připojen. Na základě zaznamenaných IP toků vytváří NetFlow statistiky, které poté předá NetFlow kolektoru. NetFlow kolektor by měl disponovat velkou úložnou kapacitou, neboť musí uchovávat v databázi statistiky z několika NetFlow exportérů.

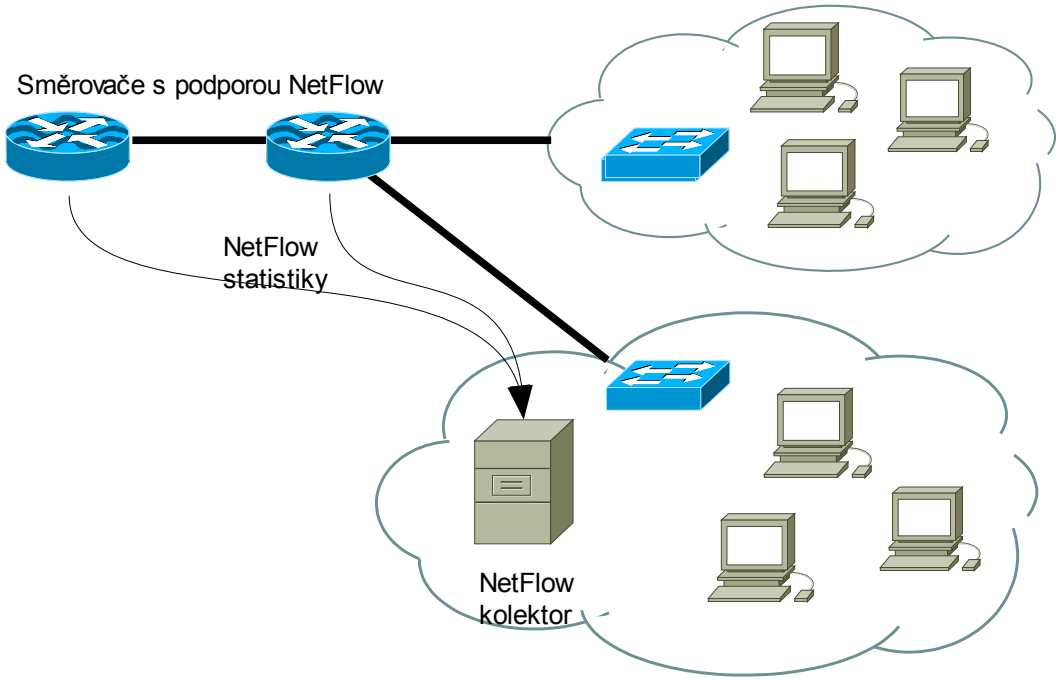
Pro zobrazení statistik je nakonec nutné použít aplikační nadstavbu, která tyto statistiky předá administrátorovi sítě v přehledné podobě. Například ve formě grafů, nebo tabulek.

Tradiční architektura

Podle společnosti Cisco Systems se u tradiční architektury na místě exportérů předpokládají směrovače, které mimo svou hlavní činnost také provádějí výpočty NetFlow statistik. Tato tradiční architektura zobrazená na obrázku 1.2 má ale několik nevýhod. Jedna z nich je například vysoká cena směrovače, který tyto výpočty dovede provádět. Proto se nevyplatí takový směrovač pořizovat do menších a středních sítí.

Druhou a podstatně důležitější nevýhodou je, že výpočty NetFlow statistik omezují směrovací výkon. To se částečně kompenzuje odebíráním každého N-tého vzorku ze vstupního toku dat do směrovače a z nich se poté vytváří NetFlow statistiky.

Popsaným řešením se ovšem sníží přesnost měření a dochází také ke snížení pravděpodobnosti odhalení bezpečnostních incidentů.

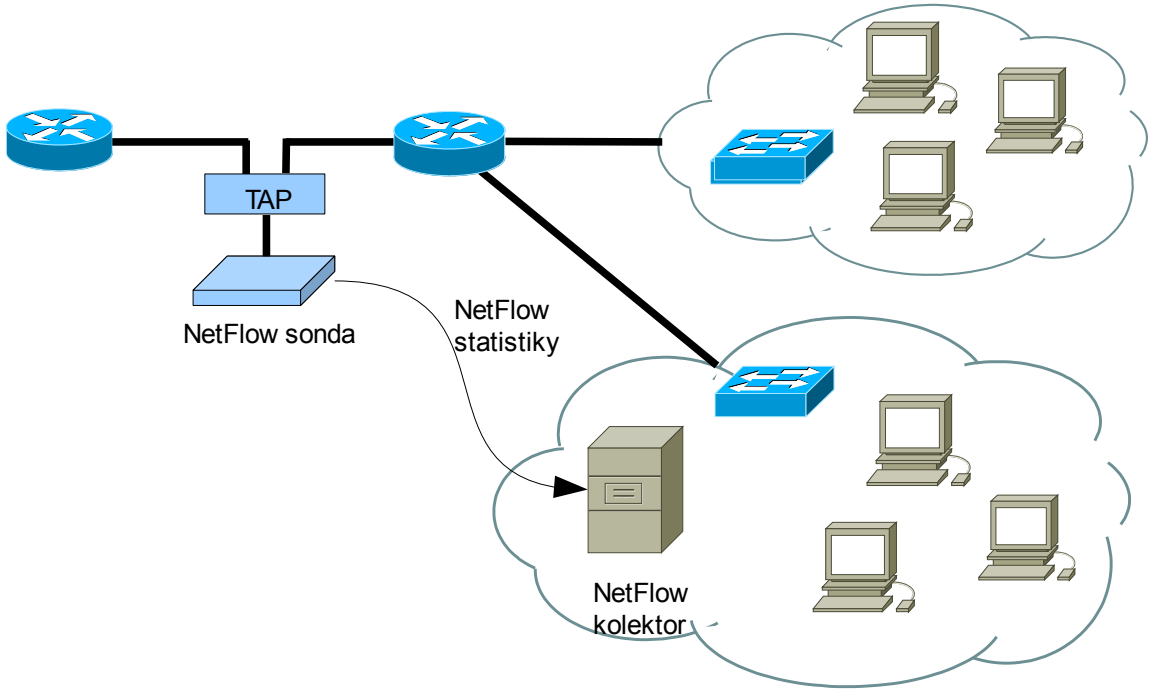


Obr. 1.2: Tradiční architektura pro NetFlow

Moderní architektura

U moderní architektury (obr. 1.3) se využívají pasivní NetFlow sondy. NetFlow sondy jsou zařízení, které vylučují všechny nevýhody tradiční architektury. Jsou to zařízení, které je možné připojit k jakémukoliv bodu v síti a to transparentním způsobem. To znamená, že NetFlow sonda do procházejících dat nijak nezasahuje. Procházející data pouze sleduje

a vytváří z nich NetFlow statistiky. Tyto statistiky jsou pak odeslány jinou vyhrazenou linkou na NetFlow kolektor, kde jsou uloženy. Proto jsou tyto data se statistikami na sledované lince zcela neviditelné.



Obr. 1.3: Moderní architektura pro NetFlow

1.2.2 NetFlow paket

NetFlow pakety se statistikami, které jsou ve směrovačích nebo v NetFlow sondách vytvořeny jsou na NetFlow kolektor přenášeny pomocí UDP nebo SCTP (Stream Control Transmission Protocol) protokolu. Jakmile je paket vyslán, tak jsou data v něm obsažené směrovačem nebo NetFlow sondou zahozeny. V případě jeho nedoručení na NetFlow kontrolér je toto bráno jako ztráta paketu a není možné tento paket znovu odeslat. Odesílání NetFlow paketů probíhá obvykle na portech 2055, 3000-3010, 9555 nebo 9995.

Paket NetFlow obsahuje:

- číslo verze NetFlow,
- sekvenční číslo,
- SNMP index vstupního a výstupního rozhraní,
- čas začátku a konce IP toku,
- počet bajtů a paketů v toku,

- údaje z L3 hlavičky:
    - zdrojové a cílové IP adresy,
    - zdrojové a cílové porty,
    - IP protokol,
    - typ služby,
- u TCP (Transport Control Protokol), toků obsahuje množinu všech TCP příznaků, které se v toku vyskytly,
- směrovací informace:
    - IP adresa příštího skoku (důležité pro analýzu směrovacích postupů),
    - maska cílové a zdrojové IP adresy (délky prefixů dle notace CIDR - Classless Inter-Domain Routing).

Některé NetFlow exportéry do přenášeného paketu uvádějí i hodnotu zdrojového a cílového autonomního systému.

1.3 Další protokoly pro sledování komunikace v síti

Existuje i řada dalších protokolů pro sledování a sbírání statistických dat o síťové komunikaci ze síťové infrastruktury. Níže jmenované protokoly jsou založeny na podobném principu jako protokol NetFlow od Cisco Systems:

- Jflow a cflowd pro Juniper Networks
- NetStream pro 3Com a HP
- NetStream pro Huawei Technology
- Cflowd pro Alcatel-Lucent
- Rflow pro Ericsson
- AppFlow pro Citrix

1.4 Srovnání výhod a nevýhod protokolu SNMP a NetFlow z hlediska jednoduchosti nasazení na sledovanou síť

1.4.1 Výhody protokolu SNMP

Protokol SNMP se stal nejrozšířenějším protokolem pro sledování síťové infrastruktury, neboť je podporován v mnoha zařízeních od různých výrobců připojitelných do počítačové sítě.

Za další výhodu protokolu SNMP lze považovat, že není nutno přidat do sítě hardware zajišťující sledování provozu. Plánuje-li se například sledování provozu sítě, je nutné pouze povolit SNMP protokol na sledovaných aktivních prvcích v síti. Sběr dat, jejich ukládání do datového úložiště a pozdější interpretace bývá zajištěna pouze programově.

Z výše uvedeného plyne, že implementace metody pro získávání statistických dat pomocí protokolu SNMP ze síťové infrastruktury je i finančně výhodná.

1.4.2 Nevýhody protokolu SNMP

Mezi hlavní nevýhody SNMP protokolu patří zatížení sítě při získávání sledovaných statistických dat. Proto je vhodné data sbírat periodicky a délku periody zvolit rozumně. Této periodě se odborně říká polling interval.

Mezi další nevýhodu patří vysoké nároky na stanici, na které poběží dohlížecí systém, který bude statistická data sbírat a ukládat do databáze. Nárok je především kladen na velikost datového úložiště.

1.4.3 Výhody protokolu NetFlow

Jako výhodu protokolu NetFlow je nutné zmínit minimální zatížení sítě při sběru sledovaných dat o síťové komunikaci. V případě použití speciálních zařízení, tzv. NetFlow sond, společně s vyhrazenými linkami (moderní architektura), je pak zatížení sledované linky nulové.

1.4.4 Nevýhody protokolu NetFlow

Aby mohl být protokol NetFlow využit pro sledování sítě, musí zařízení v síti protokol NetFlow podporovat. Protokol podporují zařízení firmy Cisco Systems.

Také je možné na sledované linky umístit NetFlow sondy. Z této nevýhody plyne i vyšší cena metody sledovaní sítě pomocí tohoto protokolu.

1.4.5 Výběr protokolu pro diplomovou práci

Jak již bylo napsáno v předchozí kapitole, pro použití protokolu NetFlow pro sledování síťové komunikace je potřeba, aby v síti byla zařízení podporující tento protokol. To je poměrně značná nevýhoda tohoto řešení.

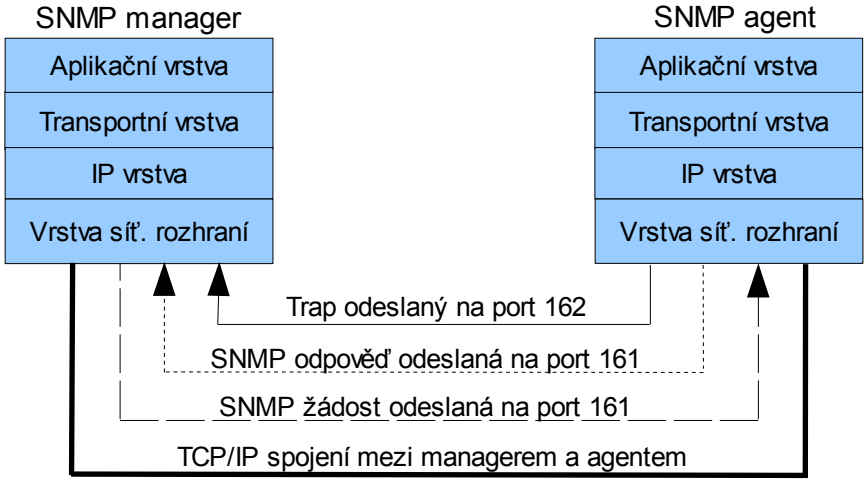
Proto je pro řešení diplomové práce použita první metoda. A to SNMP protokol, kdy je snahou minimalizovat zatížení sítě při sběru statistických dat o síťové komunikaci mezi aktivními prvky síťové infrastruktury.

2 Komunikační protokol SNMP

SNMP pro komunikaci využívá protokol UDP ze sady TCP/IP [2]. UDP bylo vybráno namísto TCP, jelikož u UDP nevytváří spojení mezi managerem a agentem před přenosem požadované zprávy. Protokol UDP je také tzv. nespolehlivým protokolem, jelikož příjemce nepotvrzuje přijetí zprávy od odesílatele. Jestli byla zpráva ztracena nebo přijata, má na starost jednoduchý časový limit (tzv. *timeout*) programu SNMP managera. Manager posílá UDP SNMP požadavky na agenta a čeká na odpověď. Jak dlouho budeme čekat na odpověď záleží na nastaveném časovém limitu v programu SNMP managera. Jestliže doba od odeslání dosáhne časového limitu a manager do této doby neobdržel odpověď na svůj požadavek, znamená to, že paket byl ztracen. Poté se může požadavek vyslat znova. Kolikrát, to záleží opět na nastavení programu SNMP managera.

Jestliže se jedná o tyto pravidelné a opakovatelné žádosti SNMP managera na odpovědi od SNMP agentů, není nespolehlivost UDP tak závažným problémem. Maximálně SNMP manager neobdrží odpověď na svou žádost. Poněkud horší je to v případě zasílání trapů. Jestliže SNMP agent odešle trap a trap není přijat SNMP managerem, tak SNMP manager ani neví, že mu byl trap vyslán. Zároveň ani agent neví, že má trap poslat znova, protože jej o trapy nikdo nežádá. Jak již bylo dříve popsáno trapy jsou odesílány bez vyžádání SNMP managera.

Když SNMP manažer posílá dotaz SNMP agentovi, tak používá zdrojový a cílový port 161. Při odesílání odpovědi se používá tentýž port. Trapy jsou odesílány na port 162. U všech zařízení podporující SNMP protokol jsou tato čísla portů výchozí.

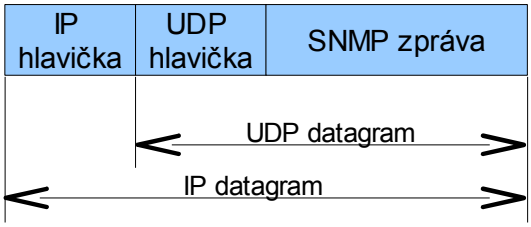


Obr. 2.1: TCP/IP komunikační model

Na obr. 2.1 je uveden TCP/IP komunikační model a naznačeno SNMP spojení. Když chce SNMP manager zažádat o nějakou hodnotu od SNMP agenta, nebo když SNMP agent chce odeslat trap, tak se dle [3] na jednotlivých vrstvách vykonává následující:

- Aplikační vrstva – První se program SNMP managera (nebo SNMP agenta) rozhodne, jakou činnost provádět. Například může zaslat SNMP žádost na agenta (nebo agent poslat odpověď na SNMP žádost, či trap).
- Transportní vrstva – Transportní vrstva umožňuje komunikovat SNMP manageru a SNMP agentovi pomocí UDP protokolu. UDP hlavička paketu obsahuje cílový port zařízení, kam je zasílána SNMP žádost nebo trap.
- IP vrstva – Tato vrstva má za úkol doručit SNMP paket příjemci, podle jeho IP adresy.
- Vrstva síťového rozhraní – Poslední vrstva, která musí SNMP paket doručit na místo určení (např. od agenta k managerovi).

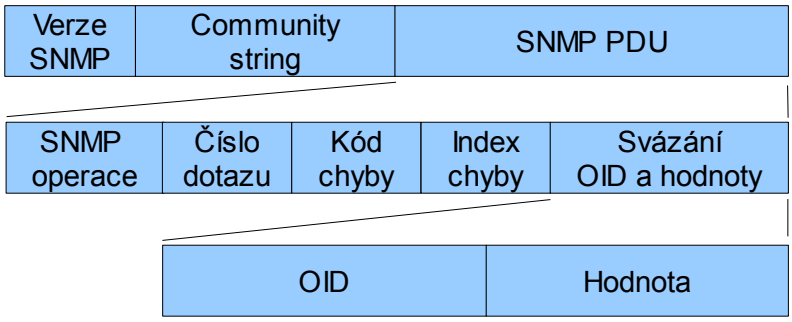
Každá SNMP zpráva je před samotným přenosem zapouzdřena do IP paketu. Tento paket je znázorněn na obr. 2.2.



Obr. 2.2: SNMP zpráva zapouzdřená do IP paketu

2.1 Formát SNMP paketu

Dle [15] je SNMP paket složen ze dvou částí. Hlavičky paketu a vlastních uživatelských dat, které jsou označovány jako PDU (Protocol Data Unit), viz obr. 2.3. Hlavička paketu obsahuje číslo verze protokolu a community string.



Obr. 2.3: SNMP paket

Vlastní datová část PDU je složena z:

- SNMP operace - jedná se o typ operace (jsou popsány v kap. 4),
- čísla dotazu - pro svázání dotazu s odpovědí,
- chybového kódu - indikuje, kde k chybě došlo a určuje její typ,
- ukazatele na objekt - přiřazuje chybu dané proměnné z pole pro svázání identifikátoru objektu (OID – Object Identifier) a hodnoty,
- svázání OID a hodnoty - přiřazuje daným proměnným jejich aktuální hodnoty (vlastní data PDU).

Datová část PDU trap pro protokol SNMPv1 je odlišná a je složena z:

- typu objektu - identifikuje typ objektu, který vygeneroval trap,
- adresy agenta – je IP adresa objektu, který vygeneroval trap,
- typu trapu,
- kódu trapu,
- časového razítka - čas mezi poslední reinicializací sítě a vygenerováním trapu,
- svázání OID a hodnoty - přiřazuje daným proměnným jejich aktuální hodnoty (vlastní data PDU).

Všechny pole PDU mohou mít proměnnou délku. Ve verzi SNMPv2 došlo ke zjednodušení a formát byl sjednocen na stejný s formátem pro ostatní operace.

3 MIB databáze

Spravované objekty, které je možné přes protokol SNMP číst nebo nastavit, jsou uloženy v databázi. Této databázi se říká Management Information Base [15]. Odtud zkratka MIB.

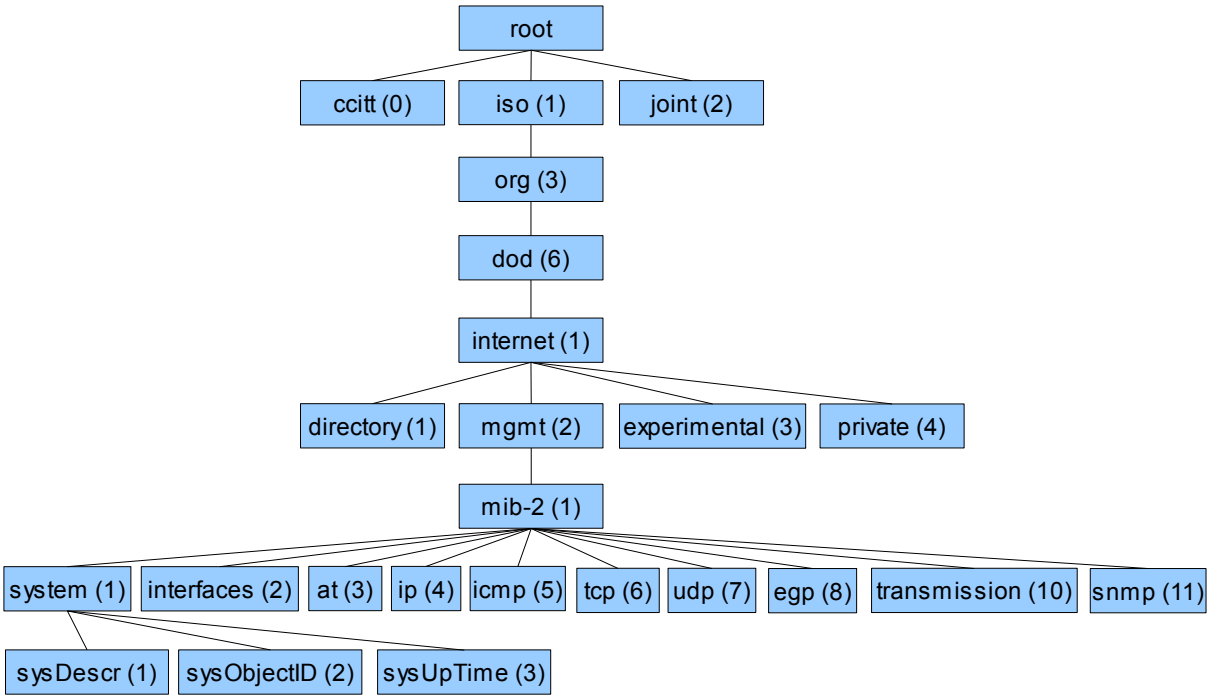
MIB má hierarchickou stromovou strukturu. Na kořen jsou připojeny uzly. Tyto uzly mohou být dále kořeny jednotlivých podstromů. Každý uzel má své jméno. Toto jméno je složeno jak z textového řetězce, tak také z čísla.

MIB databáze je tvořena z několika modulů (tříd). Jednotlivé moduly obsahují vždy skupinu souvisejících údajů.

Tyto moduly jsou uloženy jako textové soubory, které mají přesně definovanou strukturu, aby SNMP manager byl schopný informace o sledovaném zařízení v síti získat a SNMP agent tyto informace předat. Používá se notace SMI (Structure of Management Information), která je součástí ASN.1 (Abstract Syntax Notation One). S příchodem protokolu verze SNMPv2 se základní notace SMIv1 (SMI verze 1) rozšířila o několik možných spravovaných objektů na verzi SMIv2.

3.1 Přístup k objektu v MIB databázi

Pro přístup k objektu jsou používány celá čísla, které jsou odděleny tečkou. Této sekvenci čísel se říká Object Identifier – OID [3], [5]. Na obr. 3.1 je uvedena část struktury MIB databáze dle RFC 1213.



Obr. 3.1: Část struktury MIB databáze dle RFC 1213

Např. pro zjištění popisu sledovaného zařízení bude OID v číselném vyjádření:

```
.1.3.6.1.2.1.1.1
```

Dotaz SNMP agentu je možné poslat i ve formě textového řetězce. Pak by dotaz zněl:

```
.iso.org.dod.internet.mgmt.mib-2.system.sysDescr
```

Nyní bude podle obr. 3.1 popsáno, co definují jednotlivé podstromy:

- system – 1.3.6.1.2.1.1 – Definuje seznam objektů, které se týkají fungování systému jako je např. doba běhu systému, kontakt na správce zařízení, jméno zařízení, apod.
- interfaces – 1.3.6.1.2.1.2 – Uchovává informace o stavu jednotlivých rozhraních zařízení. Jestli je rozhraní ve vypnutém (anglicky down) nebo zapnutém stavu (anglicky up). Umožňuje také sledovat, kolik oktetů bylo přeneseno přes jednotlivá rozhraní v obou směrech, apod.
- at – 1.3.6.1.2.1.3 – Je podstrom pro překlad adres. Již se nepoužívá.
- ip – 1.3.6.1.2.1.4 – Uchovává mnoho záznamů souvisejících s IP protokolem. Zahrnuje taktéž směrovací tabulky.
- icmp – 1.3.6.1.2.1.5 – Obsahuje záznamy o chybách ICMP protokolu.
- tcp – 1.3.6.1.2.1.6 – Obsahuje záznamy o stavu TCP spojení.
- udp – 1.3.6.1.2.1.7 – Obsahuje statistiky UDP provozu.
- egp – 1.3.6.1.2.1.8 – Obsahuje záznamy o statistikách protokolu EGP.
- transmission – 1.3.6.1.2.1.10 – Zatím neobsahuje žádné definice.
- snmp – 1.3.6.1.2.1.11 – Zaznamenává statistiky o provozu SNMP. Umožňuje například sledovat počet odeslaných a přijatých SNMP paketů.

3.2 Rozdělení MIB

Samotná MIB databáze se dělí na dvě části. Jedná se o:

- standard MIB,
- enterprise MIB.

Standard MIB obsahuje popis objektů, které jsou univerzální pro všechny výrobce síťových zařízení. Druhá část MIB databáze, enterprise MIB, je tvořena podstromy jednotlivých výrobců síťových zařízení. Každý výrobce si tak může za dodržení notace SMI napsat svou vlastní MIB, která bude obsahovat rozšíření běžných OID [15].

3.3 Notace SMI

V definici notací SMI (Structure of Management Information) je přesně uvedeno, jak mají být objekty spravovány a jsou v ní určeny jejich datové typy. SMIv2 pak přináší rozšíření pro SNMPv2 [3].

3.4 Popis notace SMIv1

SMIv1 je definována v RFC 1155 [15]. Definici spravovaných objektů můžeme rozdělit na tři části:

- jméno,
- datový typ a syntax,
- kódování.

Jméno je identifikátor objektu (OID). Jak již bylo dříve popsáno, OID jednoznačně definuje spravovaný objekt. OID se běžně vyskytuje ve dvou formách. Může být zadáno v numerickém nebo textovém formátu.

Datový typ a syntaxe je definována podmnožinou ASN.1 (Abstract Syntax Notation One). ASN.1 je způsob, jak jsou data zastoupena pro komunikaci mezi SNMP managerem a SNMP agentem. Notace ASN.1 je nezávislá na typu zařízení.

Kódování definuje jak je řízený objekt zakódován do řetězce oktetů přenášených transportním mediem, jako je ethernet.

3.4.1 Identifikátor objektu - OID

V následujícím textu bude podrobněji popsána problematika identifikátoru objektu související s notací SMI. V kap. 3.1 již bylo zmíněno, že spravované objekty jsou uspořádány do stromové hierarchie. Tato struktura je základem pro popis SNMP. OID se skládá z řady celých čísel oddělených tečkami. Každý uzel stromu je definován číslem. Taktéž je možné použít textové názvy jednotlivých stromů a ty oddělovat tečkami. Takový zápis je pak více čitelný. Obr. 3.1 ukazuje několik úrovní takového stromu objektů v notaci SMI.

Ve stromu objektů je hlavní uzel nazván jako *root*. Z něho jsou pak vytvořeny podstromy *ccitt(0)*, *iso(1)* a *joint(2)*. Na obr. 3.1 je vidět, že uzel *iso (1)* obsahuje další podstrom jako jediný. Zbylé dva uzly jsou uzly koncovými. Tyto dva uzly ani nesouvisí s SNMP, tudíž o nich nebude více zmiňováno. Další text bude zaměřen na podstrom *iso (1)*, *org (3)*, *dod (6)*, *internet (1)*. Tento podstrom je možné vyjádřit pomocí OID textově *iso.org.dod.internet* a číselně *1.3.6.1*.

Podstrom *directory* se v současné době nepoužívá. Podstrom *mgmt* definuje standardní sadu objektů pro správu Internetu a podstrom *experimental* je rezervován pro zkušební účely.

Objekty v podstromu *private* jsou definovány jednotlivými organizacemi a výrobci zařízení, které si definují své vlastní objekty.

Zde je příklad definice podstromu *internet* a všech jeho čtyř podstromů:

```
internet      OBJECT IDENTIFIER ::= { iso org(3) dod(6) 1 }
directory     OBJECT IDENTIFIER ::= { internet 1 }
mgmt          OBJECT IDENTIFIER ::= { internet 2 }
experimental  OBJECT IDENTIFIER ::= { internet 3 }
private       OBJECT IDENTIFIER ::= { internet 4 }
```

První řádek deklaruje *internet* jako OID *1.3.6.1*, který je definován jako podstrom z *iso.org.dod* (*1.3.6*).

Další čtyři řádky jsou podobné prvnímu. Deklarují všechny čtyři podstromy (*directory*, *mgmt*, *experimental* a *private*) jako podstromy podstromu *internet*.

3.4.2 Datové typy objektů

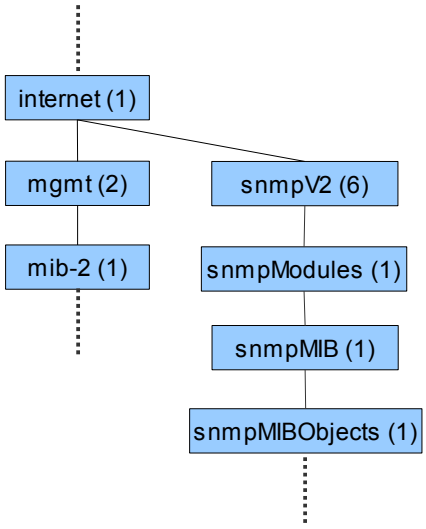
SMIv1 definuje několik datových typů, které jsou rozhodující pro management síťových zařízení. Tyto datové typy zkrátka definují, co který spravovaný objekt může obsahovat. V SMIv1 existují datové typy:

- INTEGER – Je 32 bitové číslo bez znaménka. Je často využíváno ke specifikování jednoho spravovaného objektu. Například provozní stav směrovače může ve stavu zapnutém, vypnutém, nebo testovacím. Tyto stavy mohou byt reprezentovány čísly 1 – zapnuto, 2 – vypnuto, 3 – testováno.
- OCTET STRING – Je řetězec žádného nebo více oktetů používaných k reprezentaci textového řetězce, ale někdy také k reprezentaci fyzických adres.
- Counter – 32 bitové číslo s minimální hodnotou 0 a maximální hodnotou $2^{32} - 1$. Když je dosaženo maximální hodnoty, čítač se vynuluje a začne čítat znova. Čítače se primárně užívá pro zaznamenávání informací o počtu oktetů přenesených přes dané rozhraní nebo pro počítání počtu chyb na daném rozhraní. Čítač nemůže snižovat svou hodnotu. Může ji pouze zvyšovat. Jestliže je SNMP agent restartován, hodnota čítače se nastaví na nulu.
- NULL – V současné době není v SNMP používán.

- OBJECT IDENTIFIER – Řetězec v desítkovém formátu s tečkami reprezentující spravovaný objekt ve stromě objektů. Např. *1.3.6.1.4.1.14988* reprezentuje podstrom privátních objektů pro Mikrotik.
- SEQUENCE – Definuje seznamy, které obsahují žádný nebo více jiných ASN.1 datových typů.
- SEQUENCE OF – Definuje a spravuje objekty, které jsou tvořeny sekvencemi ASN.1 typů.
- IpAddress – Používá se k reprezentaci IP adres.
- NetworkAddress – Umožňuje reprezentovat rozdílné typy síťových adres.
- Gauge – Je 32 bitové číslo s minimální hodnotou 0 a maximální hodnotou $2^{32} – 1$. Na rozdíl od čítače, Gauge umožňuje hodnotu nejen zvyšovat, ale i snižovat. Nikdy nemůže překročit jeho maximální hodnotu. Např. rychlost rozhraní směrovače se měří v jednotkách tohoto datového typu.
- TimeTicks – Je 32 bitové číslo s minimální hodnotou 0 a maximální $2^{32} – 1$. Umožňuje měřit čas v setinách sekundy. Např. doba, po kterou je zařízení v provozu, je udávána tímto datovým typem.
- Opaque – Umožňuje jiné kódování ASN.1 pro řetězce z OCTET STRING.

3.5 Rozšíření v notaci SMIv2

SMIv2 rozšiřuje strom objektů SMIv1 přidáním podstromu *snmpV2 (6)* do podstromu *internet*, jak je uvedeno na obr. 3.2. OID pro podstrom s více objekty je *1.3.6.1.6.3.1.1*, nebo-li *iso.org.dod.internet.snmpV2.snmpModules.snmpMIB.snmpMIBObjects*.



Obr. 3.2: Rozšířený strom objektů ve verzi SMIv2

Podstrom *snmpV2* přináší řadu nových datových typů:

- Integer32 – Je to samé jako INTEGER z SMIv1.
- Counter32 – Je to samé jako Counter z SMIv1.
- Gauge32 – Je to samé jako Gauge z SMIv1.
- Unsigned32 – Reprezentuje desítkové hodnoty v rozsahu 0 - $2^{32} – 1$ včetně.
- Counter64 – Je podobný jako Counter32, nicméně jeho maximální hodnota je $2^{64} – 1$. Counter64 se hodí v případech, kde Counter32 již nestačí.
- BITS – Je pro výčet nezáporných bitů.

Definice objektu v SMIv2 je mírně odlišná od definice objektu v SMIv1. Nově zde přibily volitelné položky, které není nutné vyplnit, ale umožňují větší kontrolu nad tím, jak je objekt přístupný. Také tyto položky dovolují zvětšit tabulku přidáním sloupců s lepším popisem. Syntaxe objektu v SMIv2 je následující:

```
<name> OBJECT-TYPE
    SYNTAX <datatype>
    UnitsParts <Optional, see below>
    MAX-ACCESS <See below>
    STATUS <See below>
    DESCRIPTION
        "Textual description describing this particular managed object."
    AUGMENTS { <name of table> }
    ::= { <Unique OID that defines this object> }
```

Jednotlivé položky z ukázky syntaxe objektu znamenají:

- UnitsParts – Textový popis jednotek (sekundy, milisekundy, apod.) používaných k reprezentaci objektu.
- MAX-ACCESS – Definuje typ přístupu k objektu. Platné volby jsou *read-only*, *read-write*, *read-create*, *not-accessible* a *accessible-for-notify*.
- STATUS – Tato položka byla rozšířena o klíčová slova *current*, *obsolete* a *deprecated*. *Current* v SNMPv2 je to samé jako *mandatory* v SNMPv1.
- AUGMENTS – Tato položka dovoluje rozšířit již existující tabulku jedním nebo více sloupci. Je nutné znát jméno tabulky.

SMIv2 definuje nový typ *trapu* nazývaný NOTIFICATION-TYPE a zavadí také nové textové konvence, které dovolují spravovat objekty vytvořené různými způsoby. V RFC 2579 jsou definovány nové textové konvence, které jsou v SNMPv2. Mezi ně patří:

- DisplayString – Řetězec znaků. Nesmí být delší než 255 znaků.
- PhysAddress – Fyzická adresa prezentovaná jako OCTET STRING.
- MacAddress – Definuje adresu přístupu na přenosové medium.
- TruthValue – Definuje pravda (true) a nepravda (false) hodnoty.
- TestAndIncr – Používá se ke správě dvou stanic, jestliže v nich spravujeme stejné objekty v jeden čas.
- AutonomousType – OID užitečný k definování podstromu v MIB.
- VariablePointer – je ukazatel na konkretní instanci objektu, jako je např. *ifDescr* na rozhraní *3*. V tomto případě bude VariablePointer mít OID *ifDescr.3*.
- RowPointer – je ukazatel na objekt v tabulce. Např. *ifIndex.3* ukazuje na třetí řádek v tabulce *ifTable*.
- RowStatus – Používá se pro vytváření a mazání řádků v tabulce, protože SNMP nemá vlastní způsob, jak to sám provést. RowStatus může uchovávat záznamy stavů řádků v tabulce a také přijímat příkazy pro vytvoření a mazání řádků. Textová konvence je určena na podporu integrity tabulky, když více než jeden manager aktualizuje její řádky. Stavy proměnné mohou být: *active (1)*, *notInService (2)*, *notReady (3)*, *createAndGo (4)*, *createAndWait (5)*, *anddestroy (6)*.
- TimeStamp – Měří čas, který uběhl od zapnutí systému a také zaznamenává čas některým událostem.
- TimeInterval – Měří periody času v setinách sekundy. Může nabývat hodnoty od 0 do $2^{32} – 1$.
- DateAndTime – OCTET STRING užitečný k prezentaci informace o datu a čase.
- StorageType – Definuje typ paměti, kterou SNMP agent použije.
- Tdomain – Označuje druh transportní služby.
- Taddress – Označuje adresu transportní služby. Je definován délkou 1 - 255 oktetů.

3.6 Ukázka a popis MIB souboru

Je důležité vědět, jak číst a jak rozumět MIB souborům. Proto podle [3] následuje ukázka MIB-2 souboru:

```
RFC1213-MIB DEFINITIONS ::= BEGIN

          IMPORTS
                  mgmt, NetworkAddress, IpAddress, Counter, Gauge,
                  TimeTicks
                          FROM RFC1155-SMI
                  OBJECT-TYPE
                          FROM RFC 1212;

          mib-2      OBJECT IDENTIFIER ::= { mgmt 1 }

-- podstromy v MIB-2

          system       OBJECT IDENTIFIER ::= { mib-2 1 }
          interfaces   OBJECT IDENTIFIER ::= { mib-2 2 }
          at           OBJECT IDENTIFIER ::= { mib-2 3 }
          ip           OBJECT IDENTIFIER ::= { mib-2 4 }
          icmp         OBJECT IDENTIFIER ::= { mib-2 5 }
          tcp          OBJECT IDENTIFIER ::= { mib-2 6 }
          udp          OBJECT IDENTIFIER ::= { mib-2 7 }
          egp          OBJECT IDENTIFIER ::= { mib-2 8 }
          transmission OBJECT IDENTIFIER ::= { mib-2 10 }
          snmp         OBJECT IDENTIFIER ::= { mib-2 11 }

          ifTable OBJECT-TYPE
              SYNTAX  SEQUENCE OF IfEntry
              ACCESS  not-accessible
              STATUS  mandatory
              DESCRIPTION
                  "A list of interface entries. The number of entries is
                   given by the value of ifNumber."
              ::= { interfaces 2 }
```

```
        ifEntry OBJECT-TYPE
            SYNTAX  IfEntry
            ACCESS  not-accessible
            STATUS  mandatory
            DESCRIPTION
                "An interface entry containing objects at the
                 subnetwork layer and below for a particular
                 interface."
            INDEX   { ifIndex }
            ::= { ifTable 1 }

        IfEntry ::=
            SEQUENCE {
                ifIndex
                    INTEGER,
                ifDescr
                    DisplayString,
                ifType
                    INTEGER,
                ifMtu
                    INTEGER,
                ifSpeed
                    Gauge,
                ifPhysAddress
                    PhysAddress,
                ifAdminStatus
                    INTEGER,
                ifOperStatus
                    INTEGER,
                ifLastChange
                    TimeTicks,
                ifInOctets
                    Counter,
                ifInUcastPkts
                    Counter,
                ifInNUcastPkts
                    Counter,
```

```
            ifInDiscards
                Counter,
            ifInErrors
                Counter,
            ifInUnknownProtos
                Counter,
            ifOutOctets
                Counter,
            ifOutUcastPkts
                Counter,
            ifOutNUcastPkts
                Counter,
            ifOutDiscards
                Counter,
            ifOutErrors
                Counter,
            ifOutQLen
                Gauge,
            ifSpecific
                OBJECT IDENTIFIER
        }

    ifIndex OBJECT-TYPE
        SYNTAX  INTEGER
        ACCESS  read-only
        STATUS  mandatory
        DESCRIPTION
            "A unique value for each interface. Its value ranges
             between 1 and the value of ifNumber. The value for
             each. Each interface must remain constant at least
             from one reinitialization of the entity's network-
             management system to the next reinitialization."
        ::= { ifEntry 1 }
```

```
          ifDescr OBJECT-TYPE
              SYNTAX  DisplayString (SIZE (0..255))
              ACCESS  read-only
              STATUS  mandatory
              DESCRIPTION
                  "A textual string containing information about the
                   interface. This string should include the name of
                   the manufacturer, the product name, and the version
                   of the hardware interface."
              ::= { ifEntry 2 }

END
```

První řádek tohoto souboru definuje jméno MIB. V tomto případě se jedná o RFC 1213-MIB. RFC 1213 je RFC, které definuje MIB-2. Formát této definice je vždy stejný.

V sekci IMPORTS je umožněno importovat datové typy a OID z jiných MIB souborů. V tomto příkladě je importován *mgmt*, *NetworkAddress*, *IpAddress*, *Counter*, *Gauge* a *TimeTicks* z RFC 1155-SMI a *OBJECT-TYPE* z RFC 1212.

Následně jsou identifikátory objektů OID přiřazeny jednotlivým podstromům v MIB-2. Tyto identifikátory jsou poté použity v celém souboru.

Po definování jednotlivých identifikátorů objektů se definují aktuální objekty. Každá definice objektu má následující formát:

```
<name> OBJECT-TYPE
    SYNTAX <datatype>
    ACCESS <either read-only, read-write, write-only, or not-accessible>
    STATUS <either mandatory, optional, or obsolete>
    DESCRIPTION
        "Textual description describing this particular managed object."
    ::= { <Unique OID that defines this object> }
```

Prvním definovaným objektem v příkladu je *ifTable*. Tento objekt reprezentuje tabulku síťových rozhraní, kterými disponuje spravované zařízení. Definice *ifTable* je následující:

```
ifTable OBJECT-TYPE
    SYNTAX  SEQUENCE OF IfEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "A list of interface entries. The number of entries is given by
         the value of ifNumber."
    ::= { interfaces 2 }
```

SYNTAX pro *ifTable* je *SEQUENCE OF IfEntry*. To znamená, že *ifTable* je tabulka obsahující sloupce definované v *IfEntry*. Objekt má ACCESS nastaven jako *not-accessible*. To znamená, že objekt není přístupný. Není možnost, jak se dotázat SNMP agenta na hodnotu tohoto objektu. STATUS je *mandatory*. To znamená, že je povinný a SNMP agent musí zahrnout tento objekt v souladu s MIB-2 databází. DESCRIPTION obsahuje *slovní popis*, co objekt ve skutečnosti znamená.

Unikátní OID pro *ifTable* je *1.3.6.1.2.1.2.2*, nebo-li *iso.org.dod.internet.mgmt.interfaces.2*.

V definici *ifTable* se vyskytuje typ *SEQUENCE OF*. Tato sekvence je prostý seznam objektů a jejich datových typů podle notace SMI. V tomto příkladě je očekáváno nalezení proměnných definovaných jako *ifIndex*, *idDescr*, *ifType* atd. Tento seznam může obsahovat libovolný počet řádku:

```
IfEntry ::=
    SEQUENCE {
        ifIndex
            INTEGER,
        ifDescr
            DisplayString,
        ifType
            INTEGER,
        .
        .
        ifSpecific
            OBJECT IDENTIFIER
    }
```

V definici pro *ifEntry* jsou definovány konkrétní řádky v tabulce *ifTable*. Tato definice je téměř totožná s *ifTable*. Navíc je zde zavedena položka INDEX. Tento INDEX je unikátní klíč který definuje jeden řádek v *ifTable*. Je na agentovi, aby se ujistil, že INDEX je unikátní v rámci tabulky. Jestliže má směrovač 8 rozhraní, bude v tabulce *ifTable* osm řádků.

Indentifikátor objektu *IfEntry* je *iso.org.dod.internet.mgmt.interfaces.ifTable.ifEntry*, nebo-li *1.3.6.1.2.1.2.2.1*.

```
ifEntry OBJECT-TYPE
    SYNTAX  IfEntry
    ACCESS  not-accessible
    STATUS  mandatory
    DESCRIPTION
        "An interface entry containing objects at the subnetwork layer
         and below for a particular interface."
    INDEX   { ifIndex }
    ::= { ifTable 1 }
```

INDEX pro *IfEntry* je označen jako *ifIndex* a je definován následovně:

```
ifIndex OBJECT-TYPE
    SYNTAX  INTEGER
    ACCESS  read-only
    STATUS  mandatory
    DESCRIPTION
        "A unique value for each interface. Its value ranges between
         1 and the value of ifNumber. The value for each interface
         must remain constant at least from one reinitialization of the
         entity's network-management system to the next
reinitialization."
       ::= { ifEntry 1 }
```

Objekt *ifIndex* je pouze pro čtení. To znamená, že jeho hodnota může být pouze vidět a nemůže být změněna.

Posledním objektem v ukázce je *ifDescr*. Tento objekt slouží jako slovní popis rozhraní pro každý řádek v tabulce *ifTable*.

Uvedená ukázka MIB souboru končí položkou END, která označuje konec souboru.

Komentáře v souboru je možné psát za dvě pomlčky. Ve všech případech musí být dodržována velikost písmen podle uvedeného příkladu.

4 Popis operací protokolu SNMP

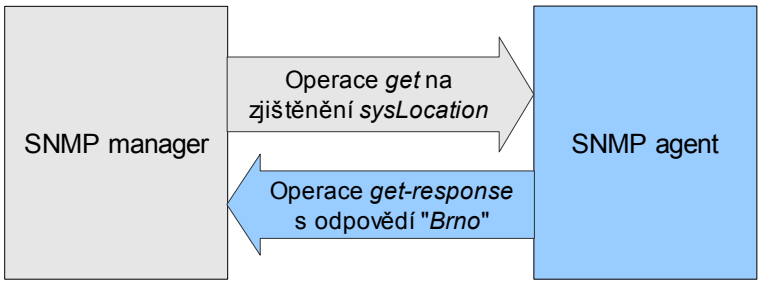
V předcházejícím textu bylo již několikrát zmíněno, že existují operace SNMP. Dle [3] mezi tyto operace patří:

- *get*
- *get-next*
- *get-bulk* (pouze v SNMPv2 a SNMPv3)
- *set*
- *get-response*
- *trap*
- *notification* (pouze v SNMPv2 a SNMPv3)
- *inform* (pouze v SNMPv2 a SNMPv3)
- *report* (pouze v SNMPv2 a SNMPv3)

4.1 Operace get

Operace *get* je operace zahajovaná SNMP managerem. Jedná se o požadavek na SNMP agenta. Agent tento požadavek zpracovává a posílá zpět odpověď k SNMP managerovi. Může však nastat situace, kdy je požadavek na straně SNMP agenta zahozen. To se může stát např. u směrovačů, které jsou provozně velmi vytížené a nezvládají požadavek *get* přijmout. Operace *get* je znázorněna na obr. 4.1.

Jednou položkou v SNMP paketu je svázání OID a hodnoty. Anglicky je toto pole označováno jako variable bindings. Toto pole je seznam MIB objektů, které dovolují na požádání zjistit, co chce SNMP manager vědět. Příjemce, SNMP agent, pak žádost vyplní (přiřadí hodnotu k určitému OID) a odešle odpověď zpět SNMP managerovi.



Obr. 4.1: Princip SNMP operace get

Na ukázku je zde uveden konzolový výpis použití operace *get*:

```
# snmpget -v 2c -c public 10.6.1.1 .1.3.6.1.2.1.1.6.0
SNMPv2-MIB::sysLocation.0 = ""
```

Při zadávání operace *get* je důležité si uvědomit, že je zadávána z SNMP managera. Pro vykonání operace *get* byl použit příkaz *snmpget*. Tento příkaz je obsažen v balíčku Net-SNMP pro unixový operační systém. Jako argument příkazu *snmpget* je zadána verze SNMP *2c*, druhý argument je pro community string (veřejný - *public*), třetí argument je IP adresa zařízení a čtvrtý identifikátor objektu *1.3.6.1.2.1.1.6.0*, na jehož hodnotu se dotazujeme.

Z kap. 3.1 je známo, že OID *1.3.6.1.2.1.1* je určen pro podstrom *system*. V dotazovaném OID jsou ale uvedeny ještě další dvě celé čísla. *6* a *0*.

6 je aktuální proměnná v MIB, na kterou je dotazováno (*sysLocation*). Jak je vidět z konzolového výpisu, tak odpověď je *system.sysLocation.0 = ""*. To znamená, že *sysLocation* není na směrovači nastaven. Také můžeme vidět, že odpověď je zachována ve formátu variable binding. *OID = hodnota*.

Na konci OID je umístěna nula. V SNMP jsou objekty MIB definovány konvencí *x.y*, kde *x* představuje aktuální OID proměnné (*1.3.6.1.2.1.1.6*) a *y* představuje identifikátor instance. Pro skalární objekty je *y* vždy *0*. Skalární objekty jsou objekty, které nejsou definovány jako řádek v tabulce. Jestliže chceme vybrat u ostatních objektů určitý řádek tabulky, tak místo nuly zapíšeme číslo řádku.

Operace *get* je užitečná pro získávání jednoho MIB objektu v jeden určitý časový okamžik. Jestliže v jeden časový okamžik je požadováno ze spravovaného zařízení získat více objektů, je vhodné použít SNMP operaci *get-next*.

4.2 Operace get-next

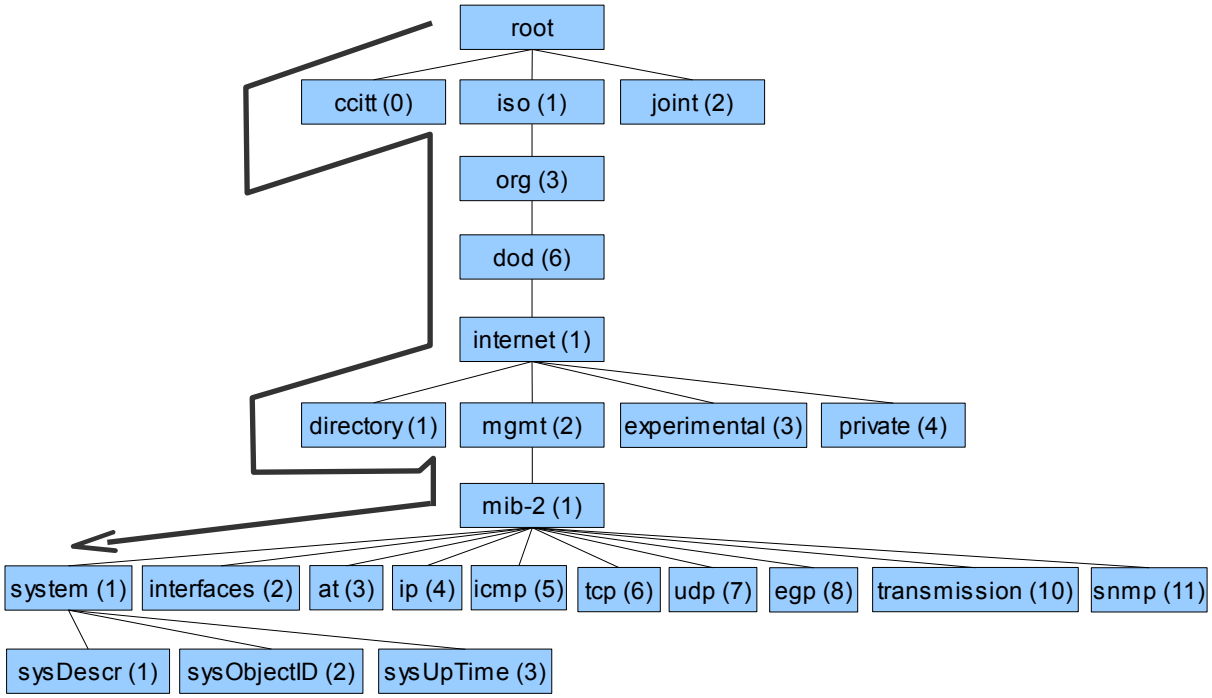
Operací *get-next* je možné načíst více hodnot z MIB. Operace *get-next* prochází podstrom postupně. Jelikož OID je sekvence celých čísel, tak je pro SNMP agenta nejjednodušší začít hledat u kořene stromu a postupně vyhledat potřebný OID. Tato metoda vyhledávání je odborně označována jako vyhledávání do hloubky. Když SNMP manager příjme odpověď od SNMP agenta na požadavek *get-next*, vyšle manager další požadavek *get-next*. Tuto činnost provádí tak dlouho, dokud mu není od SNMP agenta vrácena chyba, že nebylo dosaženo konce MIB a neexistují další objekty, které by byly zaslány s odpovědí.

Postupné zadávání operace *get-next* umožňuje příkaz *snmpwalk*. Zde je uvedena ukázka jeho použití:

```
# snmpwalk -v 2c -c public 10.6.1.1 .1.3.6.1.2.1.1
SNMPv2-MIB::sysDescr.0 = STRING: RouterOS RB532A
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.14988.1
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (93551600) 10 days,
19:51:56.00
SNMPv2-MIB::sysContact.0 = STRING:
SNMPv2-MIB::sysName.0 = STRING: Msafar
SNMPv2-MIB::sysLocation.0 = STRING:
SNMPv2-MIB::sysServices.0 = INTEGER: 78
```

Příkaz *snmpwalk* vrátil sedm MIB proměnných. Každá proměnná je částí podstromu *system* (dle RFC 1213). Je zde například vidět ID zařízení, jak dlouho je zařízení v provozu, kontaktní osobu apod.

Aby byl nalezen podstrom *system*, OID *1.3.6.1.2.1.1*, musíme začít prohledávat od kořene stromu a postupovat směrem do hloubky (dolů). Obr. 4.2 znázorňuje logický postup od kořene stromu až k podstromu *system*. U každého uzlu stromu je navštíven podstrom s nejnižším číslem. Tzn., že když po *root* uzlu následuje navštívení uzlu *ccitt*. Tento uzel však nemá žádné



Obr. 4.2: Hledání podstromu system pomocí příkazu snmpwalk

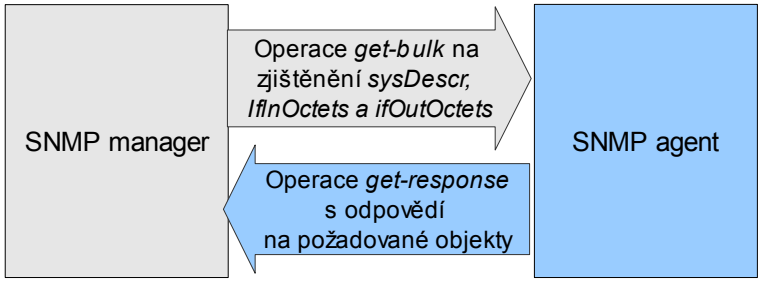
uzly pod ním, tak se vyhledávání přesune na uzel *iso*. Jelikož tento uzel má poduzly, přesune se na uzel *org*. Tento postup pokračuje až do doby, než je nalezen uzel *system*.

4.3 Operace get-bulk

V SNMPv2 je definována *get-bulk* operace, která umožňuje SNMP managerovi, aby získal velkou část tabulky najednou. Operace *get-bulk* žádá SNMP agenta o zaslání co nejvíce odpovědí. To znamená, že na některé žádosti nemusí ani odpovědět. Při zadávání příkazu musí být nastavena dvě pole. *Nonrepeaters* a *max-repetitions*.

Nonrepeaters říká, že prvních *N* objektů může být vyvoláno pomocí jednoduché operace *get-next*.

Max-repetitions říká *get-bulk* operaci, že se má pokusit o *M get-next* operací k načtení zbývajících objektů. Na obr. 4.3 je znázorněna *get-bulk* operace.



Obr. 4.3: Princip SNMP operace get-bulk

Podle obr. 4.3 jsou požadovány tři proměnné: *SysDescr*, *ifInOctets* a *ifOutOctets*. Celkový počet proměnných, které požadujeme, je dán jednoduchým vzorcem *N + (M * R)*, kde *N* je číslo pole *nonrepeaters* (skalární objekt – v tomto případě jeden – *sysDescr*), *M* je *max-repetitions* (je nastaven na 3) a *R* je počet neskalárních objektů v žádosti, či-li 2 (*ifInOctets* a *ifOutOctets*).

Jestliže je do vzorce dosazeno, je výsledek 1 + (3 * 2) = 7. Právě sedm proměnných bude vráceno v odpovědi na žádost příkazu *get-bulk.*

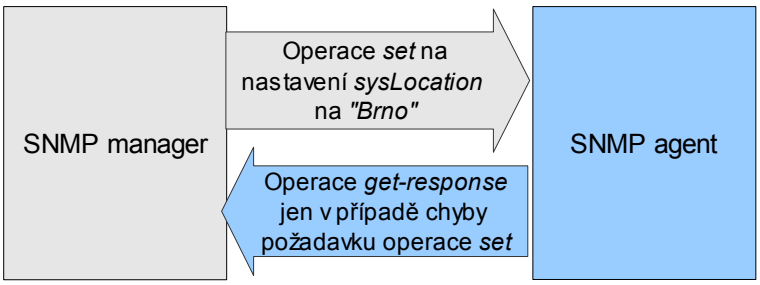
```
# snmpbulkget -v2c -c public -Cn1 -Cr3 10.6.1.1 sysDescr ifInOctets
ifOutOctets
SNMPv2-MIB::sysDescr.0 = STRING: RouterOS RB532A
IF-MIB::ifInOctets.1 = Counter32: 0
IF-MIB::ifInOctets.2 = Counter32: 1205496380
IF-MIB::ifInOctets.3 = Counter32: 472551178
IF-MIB::ifOutOctets.1 = Counter32: 0
```

```
IF-MIB::ifOutOctets.2 = Counter32: 1053176659
IF-MIB::ifOutOctets.3 = Counter32: 2458594962
```

Jak je vidět z konzolového výpisu, je v příkladu opravdu použita druhá verze SNMP (argument *–v2c*). *Nonrepeaters* a *max-repetitions* jsou nastaveny argumenty *-Cn1* a *-Cr3***.**

4.4 Operace set

Operace *set* se používá pro změnu hodnoty proměnné spravovaného objektu nebo pro vytvoření nového řádku v tabulce. Proměnné jsou definovány v MIB jako *read-write* nebo *write-only* a mohou být změněny nebo vytvořeny pomocí této operace. Pomocí této operace je také možné, aby manager nastavil více objektů najednou v jeden časový okamžik. Jestliže nastavení jednoho objektu selže, tak selže nastavení všech a žádné změny vyvolávané touto operací nebudou uskutečněny.



Obr. 4.4: Princip SNMP operace set

Obr. 4.4 ukazuje použití operace *set* a na následujícím výpisu můžeme vidět ukázku jejího použití:

```
# snmpget -v 2c -c public 10.6.1.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = ""
# snmpset -v 2c -c private 10.6.1.1 system.sysLocation.0 s "Brno"
SNMPv2-MIB::sysLocation.0 = "Brno"
# snmpget -v 2c -c public 10.6.1.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = "Brno"
```

Nejprve je pomocí příkazu *snmpget* (operace *get*) vypsáno, jak je *sysLocation* nastaven. Z odpovědi SNMP agenta je vidno, že nastaven není. Poté je zadán příkaz *snmpset* s parametry *s* a *"Brno"* (operace *set*). Parametr *s* říká, že proměnná *sysLocation* bude nastavena jako řetězec. Nastavovaný řetězec je pak napsán mezi uvozovky. To, že *sys.Location* vyžaduje řetězec je určeno podle definice *sysLocation* v RFC 1213:

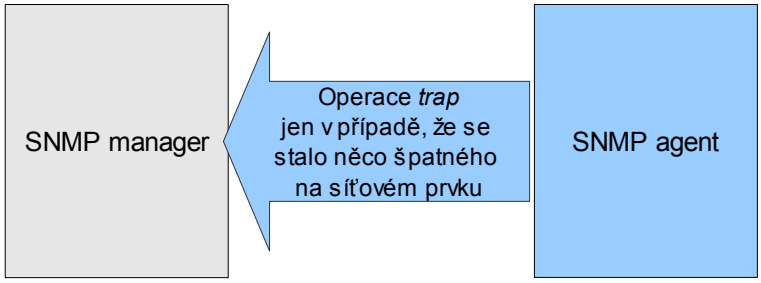
```
sysLocation OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The physical location of this node (e.g., 'telephone closet,
         3rd floor')."
    ::= { system 6 }
```

SYNTAX pro *sysLocation* je *DisplayString (SIZE (0..255))*. To znamená, že se může jednat o řetězec s maximální délkou 255 znaků.

Nakonec je v příkladu opět použit příkaz *snmpget*, aby bylo ověřeno, že hodnota proměnné *sysLocation* je opravdu nastavena.

4.5 Operace trap

Trap je způsob, kterým SNMP agent oznamuje SNMP managerovi chybové hlášení ze sledovaných zařízení. Obr. 4.5 znázorňuje operaci *trap*.



Obr. 4.5: Princip SNMP operace trap

Trap je SNMP agentem posílán na adresu, která je v agentovi nastavena. Ve většině případů to je IP adresa SNMP managera. Jestli byl *trap* managerem přijat, není agentovi potvrzováno.

Nicméně to, že se *trapy* mohou ztratit, je nečinní neužitečnými. Jsou velmi důležité pro správu sítě. Vždy je lepší vědět, že se spravované zařízení aspoň bude snažit dát vědět, že se s ním děje něco špatného, než kdyby se o podání informace vůbec nepokusilo.

Trapy mohou například hlásit, že:

- síťové rozhraní na zařízení se vypnulo,
- síťové rozhraní na zařízení se znova zapnulo,
- ventilátor směrovače se pokazil apod.

Když SNMP manager přijme *trapy*, musí vědět, jak je interpretovat. Tím je myšleno, že potřebuje vědět, co *trap* znamená a jak tento *trap* podat správci sítě. *Trap* je definován číslem. Existuje sedm základních *trapů*:

- *coldStart (0)* – Indikuje, že agent byl restartován. Všechny spravované proměnné byly resetovány (Couter, Gauge jsou nastaveny na nulu).
- *warmStart (1)* – Znamená, že se agent znova inicializoval. Spravované proměnné nebudou resetovány.
- *linkDown (2)* – Je zaslán, když se síťové rozhraní vypne.
- *linkUp (3)* – Je zaslán, když se síťové rozhraní znova zapne.
- *authenticationFailure (4)* – Je zaslán, když se někdo pokusí připojit na agenta se špatným *community stringem*.
- *egpNeighborLoss (5)* – Je zaslán, když sousední uzel využívající EGP protokol byl vypnut.
- *enterpriseSpecific (6)* – Je definován výrobcem síťového zařízení. Dále jsou definovány v podstromu enterprises v MIB databázi. Například pro Mikrotik je to podstrom *iso.org.dod.internet.private.enterprises.mikrotik*.

4.6 Operace notification

Ve snaze standardizovat PDU formát *trapu* v SNMPv1, SNMPv2 definuje *NOTIFICATION-TYPE*. PDU formát pro *NOTIFICATION-TYPE* je stejný jako pro operaci *get* a *set*. V RFC 2863 je nově definován *linkDown trap:*

```
linkDown NOTIFICATION-TYPE
    OBJECTS { ifIndex, ifAdminStatus, ifOperStatus }
    STATUS  current
    DESCRIPTION
        "A linkDown trap signifies that the SNMPv2 entity, acting in an
         agent role, has detected that the ifOperStatus object for one
         of its communication links left the down state and transitioned
         into some other state (but not into the notPresent state). This
         other state is indicated by the included value of ifOperStatus."
    ::= { snmpTraps 3 }
```

Identifikátor objektu je *1.3.6.1.6.3.1.1.5.3*, či-li *iso.org.dod.internet.snmpV2.snmpModules .snmpMIB.snmpMIBObjects.snmpTraps.linkDown.*

4.7 Operace inform

Operace *inform* je zahrnuta ve verzi SNMPv2 a SNMPv3. Poskytuje informační mechanismus, který umožňuje zasílání potvrzení o odeslaných trapech. Umožňuje také vzájemnou komunikaci dvou SNMP managerů.

4.8 Operace report

Operace *report* byla navrhnuta pro SNMPv2, ale nikdy do něj nebyla zahrnuta. V současné době je zahrnuta v SNMPv3. Tato operace má za úkol v čas informovat o případných problémech SNMP zpráv.

4.9 Hlášení o chybě pro operace get, get-next, get-bulk a set

Odpovědí s chybovým hlášením mohou pomoci určit, jak správně zadat žádost operací *get*, nebo *set*, aby byla SNMP agentem správně zpracována. Operace *get*, *getnext*, *getbulk* a *set* mohou vracet tyto chybové hlášky (stav chyby je pro každou hlášku uveden v závorce):

- *noError (0)* – Nebyl problém vykonat žádost SNMP managera.
- *tooBig (1)* – Odpověď na žádost je příliš velká na to, aby se vešla do jedné odpovědi.
- *noSuchName (2)* – Agent byl zažádán o OID, které nenalezl nebo neexistuje.
- *badValue (3)* – *Read-write* nebo *write-only* objekt byl nastaven nesprávnou hodnotou.
- *readOnly (4)* – Není používán.
- *genErr (5)* – Popisuje neznámou chybu, které neodpovídá ani jedna z předchozích.

V SNMPv2 jsou chybové hlášky rozšířeny na:

- *noAccess (6)* – Pokus o nastavení nepřístupné proměnné.
- *wrongType (7)* – Objekt byl nastaven na datový typ, který je odlišný od jeho definice.
- *wrongLength (8)* – Hodnota objektu byla nastavena na jinou hodnotu, než byla požadována.
- *wrongEncoding (9)* – Příkaz set se pokusil použít nesprávného kódování při nastavování objektu.
- *wrongValue (10)* – Pokus o nastavení hodnoty, která je nesprávná.
- *noCreation (11)* – Byl proveden pokus nastavit neexistující proměnnou nebo vytvořit proměnnou, která neexistuje v MIB.
- *inconsistentValue (12)* – Proměnná z MIB je v nekonzistentním stavu a není přijímána příkazem set.

- *resourceUnavailable (13)* – Nejsou k dispozici systémové prostředky pro provedení příkazu.
- *commitFailed (14)* – Obecná chyba po selhání příkazu set.
- *undoFailed (15)* – Příkaz nastavení selhal a agent nebyl schopný uchovat již hodnoty nastavené předem v tomtéž příkazu.
- *authorizationError (16)* – SNMP příkaz nebyl autentizován. Tzn. byl použit špatný community string
- *notWritable (17)* – Proměnná nepřijala nastavení.
- InconsistentName (18) – Byl proveden pokus o nastavení proměnné, ale pokus selhal, protože proměnná byla v nekonzistentním stavu.

5 Hotová řešení pro sběr dat o síťové komunikaci

Pro sledování síťové infrastruktury existuje v současné době několik programů, které tuto činnost umožňují [12], [14], [16]. Některé z nich dokáží přes své uživatelské rozhraní taktéž provádět nastavení aktivních prvků v síti (k tomu se využívá dříve popsaná operace SNMP set). Nicméně nedílnou součástí těchto programů jsou různé přehledy a statistiky. A to jak v textové tak i v grafické podobě.

Programy, které jsou k dohledání na Internetu, jsou většinou volně šiřitelné pod licencí GPL (General Public License), nebo to jsou také programy, které jsou komerční a musíme si je zakoupit. U nich je pak výhodou jejich komplexní řešení spočívající ve sledování síťových zařízení, jejich správu až po analýzu provozu.

V následujících kapitolách budou uvedeny některé z dostupných programů pro sledování síťové infrastruktury.

5.1 MRTG

MRTG (The Multi Router Traffic Grapher) byl vytvořen pro sledování a zaznamenávání datových toků pouze na jednom směrovači. Tento program není možné provozovat k monitorování na rozsáhlejší síti. Je jednoduchým programem, který shromažďuje sledované informace do databáze a umí z nich vytvořit grafy. Ke své činnosti používá protokol SNMP a konfiguruje se přes textový soubor.

5.2 CACTI

Cacti je v současné době nejrozšířenějším OpenSource programem vyvíjeným pod licencí GPL. K získávání sledovaných údajů používá skripty napsané v jazyce PHP. Jestliže však sledujeme větší sítě, je lepší zvolit skripty napsané v jazyce C z důvodů popsaných v kap. 6.1.

Cacti umožňuje ze sledovaných statistik, které ukládá do databáze, vytvářet různé grafy do předem připravených šablon, které jsou vytvořeny. Je možné si vytvořit i šablony vlastní a ty do Cacti importovat.

Systém Cacti nabízí i několik pokročilých funkcí jako například správu uživatelských účtů, vyhledávání zařízení v síti a jeho konfigurování, zasílání zpráv administrátorovi sítě, apod.

5.3 NAGIOS

Program Nagios se řadí mezi výkonnější NMS (Network Management System), avšak i tak je vyvíjen pod licencí GPL. Umožňuje monitorovat různé typy zařízení, např. síťové tiskárny

a umí také upozornit na chyby v síti pomocí SMS (Short Message Service) nebo e-mailu či VoIP (Voice over Internet Protocol) technologie.

Do programu Nagios je možné importovat své vlastní skripty. Tyto skripty mohou být napsány v různých programovacích jazycích. Podporovány jsou například C++, Perl, Python a PHP.

Program má jednu značnou nevýhodu. Jeho konfigurace je příliš obtížná, neboť vyžaduje programátorské znalosti správce sítě.

5.4 Komerční aplikace pro sledování sítě

Mezi nejznámější komerční aplikace pro sledování statistických údajů sítě patří:

- HP OpenView
- Cabletron Spectrum
- Solstice NetManager
- Cisco Works
- SMC EliteView
- Novell ManageWise

Tyto komerční programy jsou většinou určeny pro sledování zařízení od konkrétního výrobce. Ve většině případů nabízí funkci sledování zařízení, služeb i zdrojů, sledování, zda síť běží korektně (tzv. zdraví sítě), sbírání dat pro statistiky vytíženosti sítě, databázi síťových zařízení, zasílání upozornění administrátorovi, apod. Nejčastěji využívají protokol SNMP, avšak také umožňují použití jiných prostředků pro získání informací ze sledovaných zařízení, které SNMP protokol nepodporují.

6 Popis aplikace - praktická část

6.1 Použité programovací jazyky

Protokol SNMP je zahrnut do spousty programovacích jazyků. Například do jazyku PHP, Perl, Python, C.

Pro stěžejní části aplikace - periodický sběr statistik o síťové komunikaci a tvorbu topologie sítě nebude jazyk PHP zvolen jelikož neumožňuje odesílat více paralelních dotazů. To by se nevyplatilo v sítích, kde je řádově stovky až tisíce aktivních prvků. Například u sběru statistik o síťové komunikaci při vyslání SNMP dotazu všem těmto prvkům postupně by mohla nastat situace, kdy by nebyla obdržena SNMP odpověď ze všech těchto prvků, než by započal další okamžik pro sběr dat. Tímto by došlo ke snížení přesnosti sesbíraných statistických dat, protože sběr statistických dat by byl časově posunutý. Proto je vhodné zvolit programovací jazyk, který tyto dotazy dokáže vykonávat paralelně v několika vláknech programu. Jedná se například o jazyk Perl, Python nebo jazyk C.

Pro tvorbu skriptů, které budou mít za úkol sběr sledovaných informací ze síťových aktivních prvků byl zvolen interpretovaný programovací jazyk Python [8]. Pro tento jazyk existuje spousta knihoven pro práci s SNMP protokolem, které výrazně usnadňují práci při tvorbě programu. Operace knihoven jsou popsány v literatuře [9] a [13].

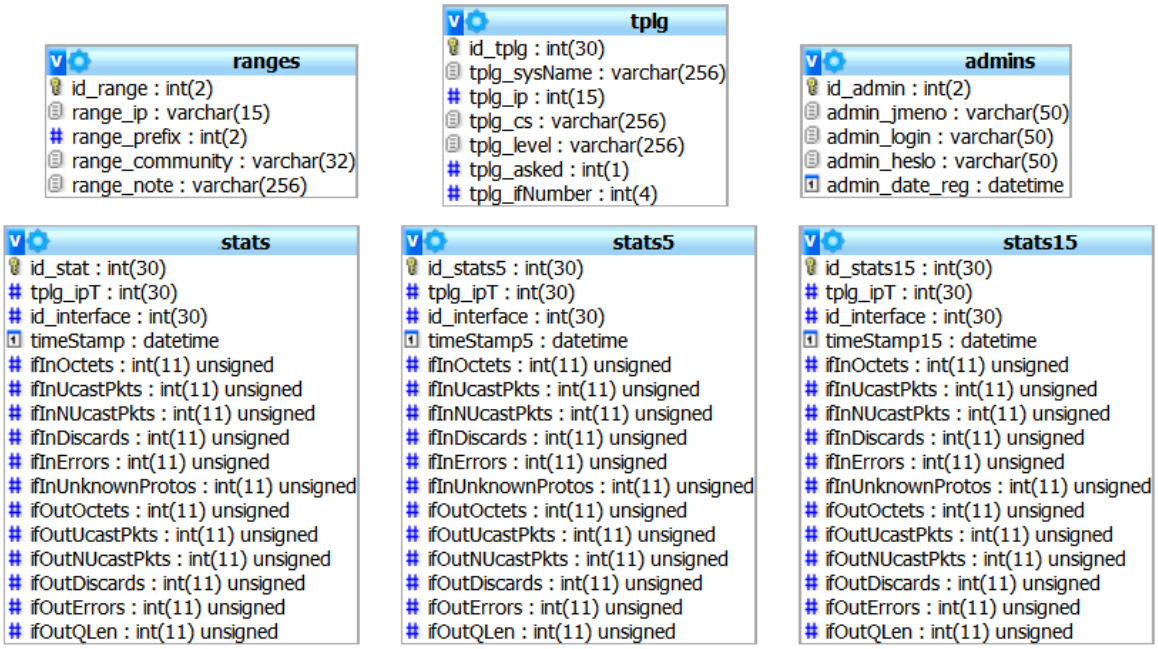
6.2 Vytvoření databáze pro uložení statistických dat

Jako databáze je použita databáze MySQL. MySQL umožňuje jednoduché propojení s jazykem Python [17], v kterém je naprogramována část programu pro sběr dat ze sítě a umožňuje také spolupráci s jazykem PHP, kterým je možné data z databáze vyčítat a následně zobrazovat administrátorovi sítě v přehledné podobě (grafy, tabulky).

V databázi je vytvořeno celkem šest tabulek podle pravidel uváděných v [4]. Na obr 6.1 je znázorněna struktura jednotlivých tabulek v databázi.

Tabulka *admins* slouží pro uchování uživatelských účtů, pomocí kterých lze přistupovat do webového rozhraní.

Tabulka *ranges* obsahuje adresu podsítě, kterou správce sítě definuje ve webovém rozhraní jako první úroveň, ze které se má vycházet při tvorbě topologie. Je uveden také prefix adresy podsítě, community string používaný v dané podsíti a taktéž poznámka správce k dané podsíti, kterou není povinné zadávat.



Obr. 6.1: Struktura tabulek v databázi MySQL

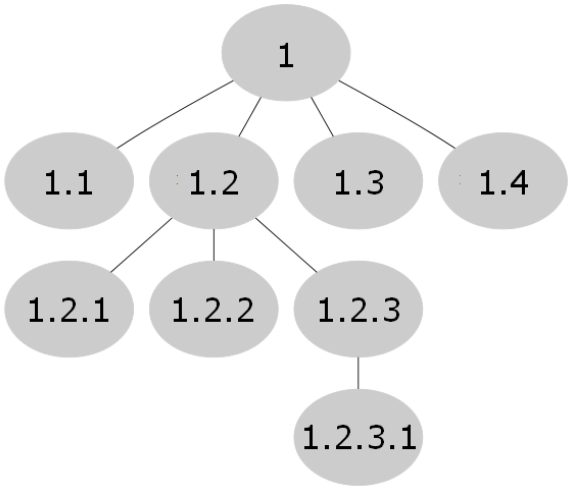
V tabulkách *stats*, *stats5* a *stats15* jsou uchovávány statistiky o síťovém provozu ze zařízení síťové infrastruktury. S každým záznamem je uveden čas zaznamenání statistiky, IP adresa a číslo rozhraní zařízení, ke kterému se daný záznam vztahuje.

Do tabulky *tplg* se ukládají všechna vyhledávaná zařízení. Z této tabulky stojí za detailní zmínku sloupec s IP adresou, *tplg_ip*, kterou reprezentuje 32 bitová celočíselná hodnota. Celočíselná hodnota z IP adresy je získána pomocí MySQL funkce *INET_ATON()*. Funkce *INET_NTOA()* zajišťuje zpětný převod na datový typ vhodný pro prezentování IP adresy (string). Dále je v tabulce sloupec uchovávající název zařízení (*tplg_sysName*), community string pro danou IP adresu, který byl odvozen z adresy rozsahu (*tplg_cs*) a sloupec, který nese hodnotu o počtu rozhraních na určitém zařízení (*tplg_ifNumber*). Zajímavými sloupci v tabulce *tplg* jsou sloupce *tplg_asked* a *tplg_level*. Sloupec *tplg_asked* slouží pro uchovávání hodnoty 1, nebo 0, při tvoření topologie a sloupec *tplg_level* pro uchování čísla úrovně, na které se zařízení v topologii nachází. O obou sloupcích je zmínka v následujících kapitolách o vytvoření topologie.

6.3 Vytvoření topologie sítě

Před vytvořením topologie, bylo mimo jiné zapotřebí se zamyslet, jak budou jednotlivé uzly spojovány při jejím vykreslování. Je vytvářen diagram, kde každému uzlu na každé úrovni je přiřazeno unikátní číslo a každému tzv. potomku rodičovského uzlu je přiděleno číslo jeho

rodiče a jeho vlastní pořadové číslo. Graf potom má strukturu čísel, jako je uvedeno na obr. 6.2.



Obr. 6.2: Struktura neorientovaného grafu

Právě tato čísla jsou ukládána do databáze MySQL k jednotlivým zařízením do sloupce *tplg_level* v tabulce *tplg*. Skripty, které tato čísla přiřazují, budou blíže popsány dále.

6.3.1 Vstupní rozsah

Adresa podsítě první úrovně zadaná přes webové rozhraní a uložená v databázi se zpracovává skriptem *zpracovani_vstup_cidr.py* napsaném v programovacím jazyce Python. Pomocí skriptu se vygeneruje soubor *list_ip_all.lg*, který tak bude obsahovat všechny hostitelské IP adresy dané podsítě:

```
10.101.0.1@public
10.101.0.2@public
10.101.0.3@public
10.101.0.4@public
...
```

Za každou IP adresu je přidáván community string, kde zavináč slouží jako delimiter.

6.3.2 První úroveň

Dalším krokem pro vytvoření topologie sítě je odeslání *ICMP echo request* (dále je používán výraz PING[3]) na všechny IP adresy vygenerované v předcházejícím kroku a uložené v souboru *list_ip_all.lg*. Tento ping je prováděn maximálně třikrát. Neodpoví-li dotazovaná

3 PING - Packet InterNet Groper - program pro ověření dostupnosti síťového rozhraní

IP adresa ani na jeden PING, je považována za nedostupnou a dále se s touto IP adresou nepracuje.

Kdyby se měl PING provádět na další IP adresu až po skončení PINGu právě prováděného, trvalo by značně dlouho, než by se manager všech IP adres o *ICMP echo reply* dotázal. Proto je ve skriptu zaveden modul, který umožňuje vláknové provádění programu. Počet vláken je v tomto skriptu omezen na padesát.

V každém vlákně je položka seznamu s IP adresou rozdělena na samotnou IP adresu a community string. Poté je proveden na danou IP adresu PING. Jestliže se vrátí *ICMP echo reply*, tak se provede SNMP dotaz, který má za úkol zjistit název zařízení. Pokud je tento příkaz bez SNMP odpovědi (vyprší čas na odpověď), znamená to, že zařízení nemá SNMP protokol aktivován. Tím se provede vytřídění zařízení, které nepodporují SNMP protokol, nebo jej nemají aktivován. Každý SNMP dotaz při vytváření topologie s SNMP zařízeními se provede maximálně třikrát.

Poté následuje dotaz k databázi MySQL. Má se zjistit, jestli se v tabulce *tplg* již nachází zařízení s IP adresou, se kterou byl v daném vlákně skriptu prováděn příkaz PING. Jestliže ano, uzavře se přístup k databázi a vlákno se ukončí.

V druhém případě, pokud daná IP adresa v tabulce *tplg* obsažena není, začne se program dotazovat, kolik má právě dotazované zařízení rozhraní. Tuto hodnotu poté uloží do databáze spolu s IP adresou, community stringem, úrovní topologie (1. úroveň) a příznakem dotazování na směrovací tabulky (hodnotou 0).

Výše uvedené řeší algoritmus skriptu *zpracovani_tplg_ICMP.py*.

6.3.3 N-tá úroveň

Pro zjištění všech dalších úrovní topologie sítě je využíván algoritmus skriptu *zpracovani_tplg_N_level.py*. Algoritmus si do proměnných typu seznam uloží z databáze všechny IP adresy, community stringy, názvy zařízení a čísla jejich úrovně v topologii.

Všem těmto zařízením je změněna hodnota *tplg_asked* v databázi na jedna, jelikož se jich bude dotazováno na nepřímé cesty ve směrovací tabulce. (Hodnota nula je pro ta zařízení v databázi, která ještě nebyla zpracována tímto skriptem. Jestliže je "1" u všech zařízení, tak byla celá síť prohledána a neexistují žádné další nepřímé cesty z právě dotazovaného směrovače.)

Poté jsou vytvořeny vlákna pro každé zařízení, ve kterých jsou uskutečněny SNMP dotazy na cílové adresy nepřímých cest ve směrovací tabulce. Tím jsou objeveny zařízení další úrovně a ty jsou opět přidávány do databáze. Nejprve se však ověří, zda se zařízení již

v databázi nevyskytuje. Do databáze je příslušnému záznamu přiřazeno také počet rozhraní, číslo úrovně a hodnota pro *tplg_asked* (nula).

SNMP dotazy jsou opět prováděny vícekrát, avšak maximálně třikrát. Jednou nejsou prováděny z toho důvodu, kdyby se SNMP dotaz zahodil při vytížené sítí z důvodu nízké priority SNMP a maximálně třikrát, což je kompromis mezi spolehlivostí a dobou vyhledaní topologie sítě.

Hledání další úrovně se provádí až po skončení vykonávání všech vláken úrovně právě vykonávané. Čekaní na skončení vykonávaných vláken je ošetřeno následujícím cyklem *for* s podmínkou *if*.

```
for thread in threading.enumerate():
    if thread is not threading.currentThread():
        thread.join()
```

6.3.4 Generování vzhledu topologie

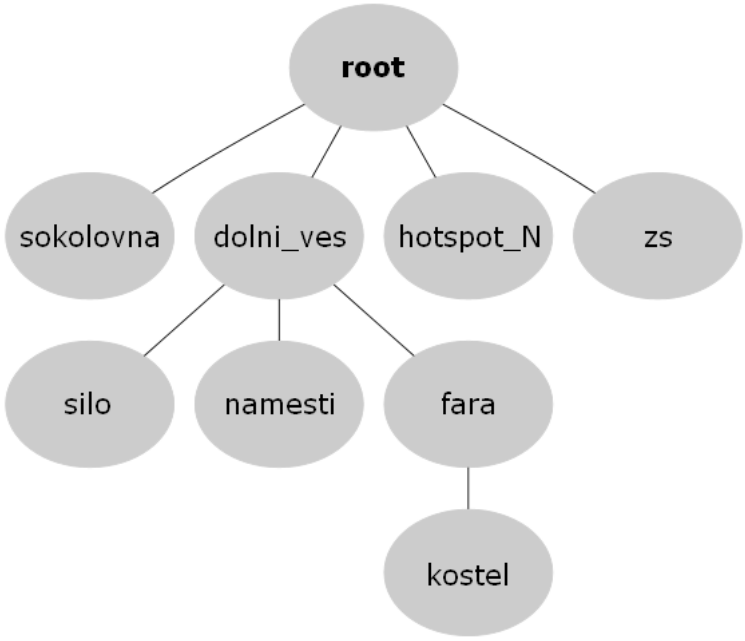
Proto, aby bylo možné zobrazit topologii sítě na základě uložených čísel úrovní, je potřeba algoritmu, který vytvoří zdrojový kód pro program GraphViz. GraphViz je nástroj pro vizualizaci grafů a diagramů běžících pod licencí Eclipse Public License [10]. Zdrojový kód je tvořen postupně. Je vytvořen soubor, do kterého je nejprve uložena hlavička programu, která definuje barvy diagramu, barvy písma, typ písma, tvary jednotlivých uzlů atd. a těla programu, ve které je definováno propojení jednotlivých uzlů.

Z databáze jsou postupně získávány názvy a IP adresy všech zařízení. Na základě jejich čísel úrovní jsou zařízení propojována k sobě. K tomu aby se nám spojily dva uzly, například uzel A a B, stačí napsat, vygenerovat kód, který ve svém těle bude obsahovat řádek *A--B*. Tzn., že pro případ topologie sítě bude použito dvou pomlček mezi uzly. Následuje ukázka zdrojového kódu, který je možné vygenerovat skriptem *dia.py*.

```
graph G {
node [shape=ellipse, style=filled,fillcolor=gray60,
color=black,fixedsize=true, height=1.5, width=2,fontcolor=black,
fontsize=14];
"root"--"10.101.0.2"
"root" [URL="../popis.php?id=0", fillcolor="gray80", target="_blank",
alt="root" ,color="gray80", label="root"];
"10.101.0.2" [URL="../popis.php?id=1", fillcolor="gray80", target="_blank",
alt="10.101.0.2" ,color="gray80", label="HS_namesti"];
```

```
"root"--"10.101.0.10"
"root" [URL="../popis.php?id=0", fillcolor="gray80", target="_blank",
alt="root" ,color="gray80", label="root"];
"10.101.0.10" [URL="../popis.php?id=2", fillcolor="gray80", target="_blank",
alt="10.101.0.10" ,color="gray80", label="skola"];
}
Pozn.: Výpis je zkrácen.
```

Tímto je zdrojový kód pro program GraphViz hotov a pozdějším přeložením souboru tímto programem vznikne vektorový obrázek formátu **.svg* ekvivalentní k obr 6.3. Tento obrázek je pak vložen do webové aplikace na místo pro zobrazení topologie. Obrázek umožňuje na jednotlivé uzly kliknout díky definování URL, jak je vidno z ukázky zdrojového kódu. Po kliknutí na určitý uzel (zařízení) ve webové prezentaci jsou zobrazeny jeho statistická data.



Obr. 6.3: Vygenerovaná topologie sítě programem GraphViz

6.4 Aktuální statistická data ze síťového zařízení

Tato část aplikace je převážně vytvořena pomocí programovacího jazyka PHP a pomocí balíčku *php5-snmp* Debian GNU/Linuxu.

Statistická data jsou ze směrovače s IP adresou *IP* zjištěny příkazem:

```
snmpwalk -v 2c -c CS IP OID
```

kde parametr *-v* určuje verzi SNMP protokolu a parametr *-c* community string *CS*. Za *OID* je dosazováno konkrétní OID, pro které má být zjištěna jeho hodnota. Tento příkaz je vykonán pomocí PHP funkce *exec()*, která na svůj výstup uloží do proměnné datového typu pole všechna variable bindings (OID: hodnota).

Následují příkazy SNMP dotazů zadávaných ve skriptu jazyka PHP *popis.php* pro získání všech základních statistických dat ze zařízení.

Získání popisných informací o zařízení je zajištěno následujícím SNMP dotazem v příkazu command:

```
$command = "snmpwalk -v 2c -c $tplg_cs $tplg_ip 1.3.6.1.2.1.1";
exec($command, $output);
```

Po zadání tohoto příkazu s SNMP dotazem budou v proměnné output variable bindings pro popis terminálu privátní OID, uptime, kontaktní informace, název terminálu, umístění terminálu a typ poskytující služby. Hodnoty mohou být například následující:

- Popis terminálu: *RouterOS RB433AH,*
- Privátní OID: *SNMPv2-SMI::enterprises.14988.1,*
- Uptime: *(336457900) 38 days, 22:36:19.00,*
- Kontaktní informace: *email@mejl.cz,*
- Název terminálu: *fara,*
- Umístění terminálu: *dolni_ves,*
- Typ poskytující služby: *78.*

Tabulka popisující rozhraní směrovače je získána pomocí následujícího příkazu:

```
$command = "snmpwalk -v 2c -c $tplg_cs $tplg_ip 1.3.6.1.2.1.2.2.1";
exec($command, $outputR);
```

Při vykonaní tohoto příkazu jsou brány v úvahu variable bindings pro index rozhraní, popis rozhraní, typ rozhraní, maximální velikost datagramu, rychlost rozhraní, MAC adresa rozhraní, administrativní status rozhraní a stav rozhraní. Hodnoty pro tabulku rozhraní mohou být například následující:

- index rozhraní: *1,*
- popis rozhraní: *ether1-hvfn11,*
- typ rozhraní: *ethernetCsmacd(6),*
- maximální velikost datagramu: *1500,*

- rychlost rozhraní: *100000000,*
- MAC adresa rozhraní: *00:0c:42:77:82:cb,*
- administrativní status rozhraní: *up (1),*
- stav rozhraní: *down (2).*

Následující příkaz je zadáván pro zjištění adresní tabulky pro jednotlivá rozhraní:

```
$command = "snmpwalk -v 2c -c $tplg_cs $tplg_ip 1.3.6.1.2.1.4.20.1";
exec($command, $outputA);
```

Jsou zjišťovány hodnoty pro IP adresu, rozhraní, maska podsítě, všesměrová IP adresu a maximální velikost datagramu. Zpětně mohou být získány například hodnoty:

- IP adresa: *10.1.5.1,*
- rozhraní: *6,*
- maska podsítě: *255.255.255.0,*
- všesměrová IP adresa: *1,*
- max. velikost datagramu: *65535.*

Za velmi důležitou tabulku lze považovat směrovací tabulku. Ta je získávána tímto příkazem:

```
$command = "snmpwalk -v 2c -c $tplg_cs $tplg_ip 1.3.6.1.2.1.4.21.1";
exec($command, $outputS);
```

Jako důležitá OID byly vybrány IP adresa cíle, index rozhraní, primární metrika, IP adresa dalšího skoku, typ trasy, směrovací protokol a maska. Získané hodnoty na daná OID mohou být následující:

- IP adresa cíle: *10.1.6.0,*
- index rozhraní: *2,*
- primární metrika: *1,*
- IP adresa dalšího skoku: *10.101.0.11,*
- typ trasy: *indirect(4),*
- směrovací protokol: *local(2),*
- maska: *255.255.255.0.*

Pomocí SNMP dotazu na tabulku Net To Media v příkazu command zjistíme připojené síťové prvky k zařízení:

```
$command = "snmpwalk -v 2c -c $tplg_cs $tplg_ip 1.3.6.1.2.1.4.22.1";
exec($command, $outputN);
```

Jsou získávány hodnoty pro OID vypovídající o index rozhraní, MAC adrese, IP adresa a typ mapování. Získané hodnoty mohou být následující:

- index rozhraní: *6,*
- MAC adresa: *00:0b:6b:2a:ce:34,*
- IP adresa: *10.1.5.114,*
- typ mapování: *dynamic(3).*

Jednotlivé položky proměnné typu pole z výstupu každého příkazu jsou rozděleny na OID a jejich hodnotu. OID je zahozeno a ve webové části prezentace nahrazeno vypovídajícím českým ekvivalentem. Ke každému ekvivalentu je pak přiřazena i hodnota daného OID. Viz např. obr. 6.4 tabulky z webového rozhraní obsahují směrovací tabulku

IP adresa cíle	Index rozhraní	Primární metrika	IP adresa dalšího skoku	Typ trasy	Směrovací protokol	Maska
10.0.0.0	2	1	10.101.0.2	indirect(4)	local(2)	255.255.0.0
10.1.2.0	3	0	10.1.2.1	direct(3)	other(1)	255.255.255.0
10.1.4.0	5	0	10.1.4.1	direct(3)	other(1)	255.255.255.0
10.1.6.0	2	1	10.101.0.11	indirect(4)	local(2)	255.255.255.0
10.1.30.0	4	1	10.1.3.207	indirect(4)	local(2)	255.255.255.0
10.1.40.0	5	1	10.1.4.203	indirect(4)	local(2)	255.255.255.0
10.6.0.0	2	1	10.101.0.60	indirect(4)	local(2)	255.255.0.0
10.13.0.0	2	1	10.101.0.60	indirect(4)	local(2)	255.255.0.0

Obr. 6.4: Směrovací tabulka z webového rozhraní aplikace SDSKSI

Po kliknutí na uzel topologie sítě se kromě těchto tabulek zobrazí i tabulka, která vypisuje počet všech přenesených oktetů na určitém rozhraní ve vysílacím i přijímacím směru. Dále je v této tabulce uveden i počet zahozených a chybných oktetů. Tato tabulka je zobrazována proto, aby správce sítě mohl tuto stránku aktualizovat například klávesou F5 a viděl, zda například počet chybných, či zahazovaných oktetů v krátkém časovém intervalu nenarůstá.

Tato statistická data všech zmiňovaných tabulek nejsou ukládána do databáze, jelikož jsou požadována vždy aktuální. Při nahlédnutí je možné tato statistická data exportovat do souboru. Více o exportu statistik je napsáno v kap. 6.6.

6.5 *Sběr statistik o síťovém provozu*

Perioda sběru dat ze síťového prvku byla stanovena na 1 minutu. Proto bylo zapotřebí udělat pro sběr statistických dat ze všech rozhraní skript, který se bude moci opakovaně spouštět. Automaticky. Tohoto automatického spouštění je dosaženo pomocí Cronu v Debian GNU/Linuxu. Záležitostem okolo Cronu se věnuje kap. 6.5.4.

Na začátku skriptu *stats.py* pro sběr statistik jsou z databáze načteny IP adresy a community stringy všech zařízení, která při vytvoření topologie byla nalezena. Poté jsou opět tvořeny vlákna, ve kterých se pro každé rozhraní daného zařízení žádá o statistiky síťového provozu. Jestliže zařízení na SNMP dotaz neodpoví, je pravděpodobně v danou chvíli nedostupné a pro tento moment jsou do databáze na všechny dotazované OID uloženy hodnoty 0. Jestliže je přijata SNMP odpověď, tak jsou do databáze navrácené hodnoty jednotlivých OID uloženy.

Do databáze se ukládají hodnoty pro OID z příchozího směru na dané rozhraní:

- ifInOctets - Celkový počet přijatých oktetů.
- ifInUcastPkts - Počet unicast paketů.
- ifInNUcastPkts - Počet ne unicast paketů (například všesměrových nebo multicastových).
- ifInDiscards - Počet paketů, které byly zahozeny.
- ifInErrors - Počet chybně přijatých paketů.
- ifInUnknownProtos - počet zahozených paketů na základě neznámého nebo nepodporovaného protolu.

A také hodnoty pro OID z odchozího směru na daném rozhraní:

- ifOutOctets - Celkový počet odeslaných oktetů.
- ifOutUcastPkts - Počet unicast paketů.
- ifOutNUcastPkts - Počet ne unicast paketů (například všesměrových nebo multicastových).
- ifOutDiscards - Počet paketů, které byly zahozeny.
- ifOutErrors - Počet chybně odeslaných paketů.
- ifOutQLen - Délka fronty na výstupním rozhraní (v paketech).

6.5.1 Průměrování statistik

Pro vytvoření grafů za posledních 24 hodin jsou používány statistiky pětiminutové. Tyto jsou vytvořeny zprůměrováním minutových statistik z tabulky *stats* a jsou uloženy do tabulky

stats5. Obdobně je tomu u statistik za poslední týden. Tyto jsou průměrovány po 15 minutách a jsou ukládány do tabulky *stats15*. O průměrování se starají skripty *stats5.py* a *stats15.py*. Například pro zprůměrovaní hodnot patnácti minutových statistik je zadán tento SQL dotaz do databáze MySQL dle požadavků v [1]:

```
SELECT ROUND( AVG( ifInOctets ) ) , \
ROUND( AVG( ifInUcastPkts ) ) , ROUND( AVG( ifInNUcastPkts ) ) , \
ROUND( AVG( ifInDiscards ) ) , ROUND( AVG( ifInErrors ) ) , \
ROUND( AVG( ifInUnknownProtos ) ) , ROUND( AVG( ifOutOctets ) ) , \
ROUND( AVG( ifOutUcastPkts ) ) , ROUND( AVG( ifOutNUcastPkts ) ) , \
ROUND( AVG( ifOutDiscards ) ) , \
ROUND( AVG( ifOutErrors ) ) , ROUND( AVG( ifOutQLen ) ), id_interface,
tplg_ipT \
FROM twelfth \
WHERE ( \
timeStampT > DATE_SUB( NOW( ) , INTERVAL 15 MINUTE ) ) \
GROUP BY id_interface, tplg_ipT
```

Z výpisu vyplývá, že jsou průměrovány veškeré hodnoty tabulky *stats5* pro každé rozhraní a IP adresu.

6.5.2 Vytvoření grafu propustnosti

Pro vytvoření grafu propustnosti konkretního rozhraní je použita knihovna JpGraph [6] vytvořená v jazyce PHP. Pro tuto knihovnu v podstatě stačí definovat pole vstupních dat a graf je pak dle dalších požadavků vykreslen.

Vstupními daty je 60 hodnot v případě grafu propustnosti za poslední hodinu. Tato data musí být vypočítána dle rovnice 1.1 [11].

$$R = \frac{\left| octets_{t+1} - octets_{t} \right|}{\Delta t} \qquad [Byte/s] \tag{1.1}$$

Jak již bylo dříve zmíněno, statistika přenesených oktetů je navyšována jako čítač. Proto se pro množství přenesených dat za poslední minutu musí od aktuální statistiky odečíst statistika předchozí a vydělit časem, který uplynul mezi dvěma odečty čítače. Rovnice 1.1 bude ve zdrojovém kódu skriptu *grag60.py* vypadat následovně:

```
rozdilIn=abs(int(octets_in[i+1])-int(octets_in[i]))
trafficIn=((rozdilIn*8)/1000)/casDelta   # *8 /1000 /> [kbps]
```

Násobení osmi a dělení tisícem je prováděno proto, aby výsledek byl v kb/s.

I čítače síťového provozu mohou přetéct. Nabývají 32 bitové hodnoty datového typu *counter32* popisovaného v kap. 3.5. Aby se při přetečení nezobrazila v grafu špička (o teoretické velikosti 2^{32}) je případ ošetřen výpočtem:

```
if (int(octets_in[i+1])>int(octets_in[i])):          #osetreni preteceni ctr
    rozdilIn=2^32-int(octets_in[i+1])+int(octets_in[i])             #vypocet
    trafficIn=((rozdilIn*8)/1000)/casDelta              # *8 /1000 /> [kbps]
```

Podobná špička může vzniknout i při restartu zařízení nebo po krátkodobém výpadku napájení, kdy se čítače automaticky vynulují. Proto je na základě znalostí rychlosti rozhraní za *trafficIN*, resp. *trafficOut*, dosazována hodnota 0.

```
if (rozdilIn*8)/1000>ifSpeed:                          #osetreni resetu ctr
    trafficIn=0                                      #vlozeni nuly misto 2^32
```

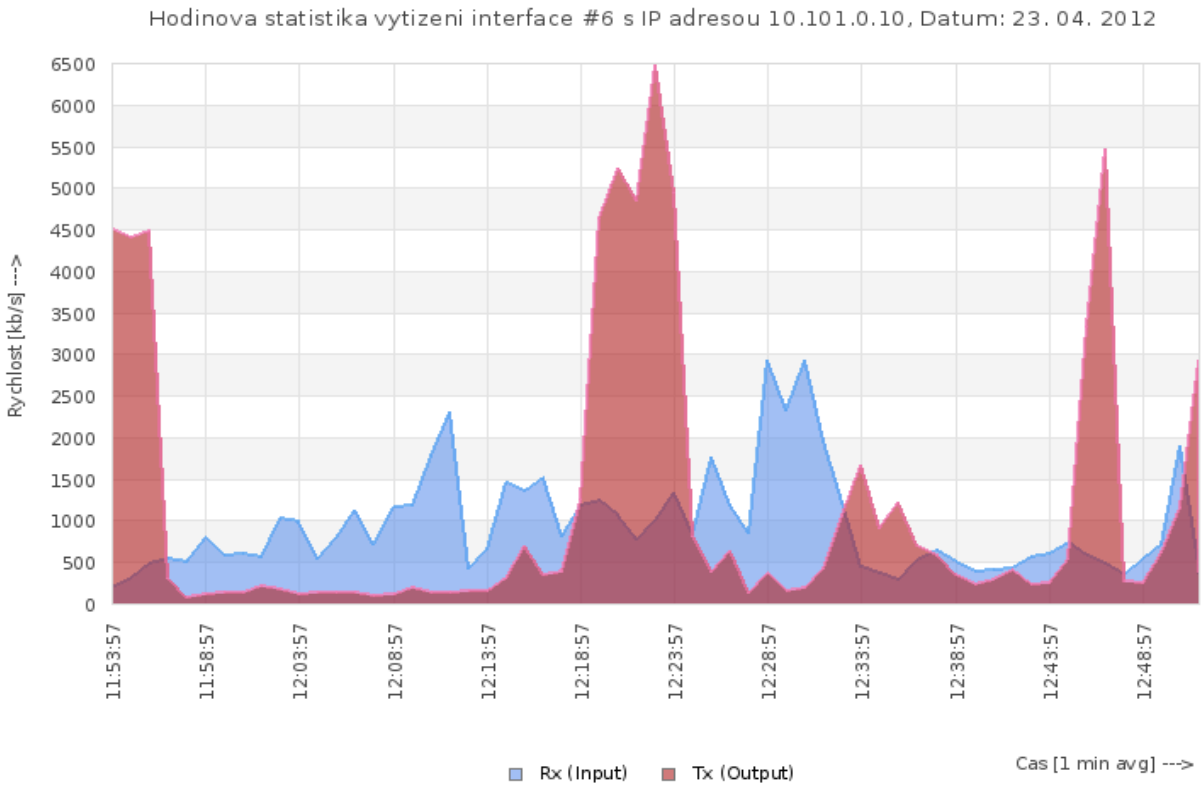
V obou případech platí, že nikdy nemůže přes rozhraní procházet větší objem dat, než na které je maximálně definováno. Vypočítané hodnoty jsou pomocí python skriptu přes vytvořený soubor (např. *hodnoty60.lg*) předávány na vstup zpracování grafů pomocí knihovny JpGraph [6].

V knihovně JpGraph je vstupní pole s hodnotami zpracováno jak ukazuje následující ukázka zdrojového kódu:

```
<?php
    require_once ('/var/www/SDSKSI/gd/jpgraph/jpgraph.php');
    require_once ('/var/www/SDSKSI/gd/jpgraph/jpgraph_line.php');
    require_once ('/var/www/SDSKSI/gd/jpgraph/jpgraph_date.php');

    //nacteni hodnot do grafu
    $dataIn = File ("/var/www/SDSKSI/python/hodnoty60In.lg");
      for ($i = 0; $i < Count ($dataIn); $i++)
        $dataIn[$i]=(float)$dataIn[$i];
    //nacteni hodnot do grafu
    $dataOut = File ("/var/www/SDSKSI/python/hodnoty60Out.lg");
      for ($i = 0; $i < Count ($dataOut); $i++)
        $dataOut[$i]=(float)$dataOut[$i];

    // Create the Rx line
    $p1 = new LinePlot($dataIn,$datax);
    $p1->SetColor("blue");
```

```
    $p1->SetLegend('Rx (Input)');
    $p1->SetFillColor("#6495ED@0.4");
    $graph->Add($p1);

    // Create the Tx line
    $p2 = new LinePlot($dataOut,$datax);
    $p2->SetColor("red");
    $p2->SetLegend('Tx (Output)');
    $p2->SetFillColor("#B22222@0.4");
    $graph->Add($p2);

    // Ulozi obrazek grafu jako obrazek png do souboru s cislem interface
    $graph->Stroke("/var/www/SDSKSI/grafy/obr/60.png");
?>
Pozn.: Výpis je zkrácen. Neobsahuje formátování grafu.
```

Je zde doplněn vzhled grafu, popis os, legenda, měřítka os atd. Následně je vygenerován obrázek v grafickém formátu **.png* jako je např obr. 6.5. Takový obrázek je pak pomocí PHP zobrazován ve webovém rozhraní aplikace.



Obr. 6.5: Graf propustnosti na rozhraní směrovače vygenerovaný pomocí JpGraph

6.5.3 Promazávání starých statistik

Jelikož je do databáze MySQL ukládán velký objem dat, tak je prováděno vyprazdňování tabulek uchovávající statistická data v databázi. Například jen při 100 síťových rozhraních několika směrovačů je za den uloženo 144 000 záznamů minutové statistiky, 28 800 záznamů pětiminutové statistiky a 9 600 záznamů patnáctiminutové statistiky.

Proto všechny minutové statistiky, které jsou starší než dvě hodiny jsou z tabulky *stats* odstraňovány každou neděli. Že se to provede právě každou neděli je zajištěno pomocí Cronu (popisuje následující kap. 6.5.4), a že se smažou jen statistiky starší než dvě hodiny, je dosaženo pomocí příkazu DELETE odeslaného do databáze MySQL:

```
sql = "DELETE FROM stats WHERE (timeStamp < DATE_SUB(NOW(),
INTERVAL 2 HOUR ))"
```

Formát příkazu SQL příkazu DELETE je popsán v literatuře [1].

Mazání statistik pětiminutových a čtvrthodinových je obdobné. U pětiminutových je mazání prováděno taktéž každou neděli a jsou smazány statistiky starší 72 hodin. Čtvrthodinové statistiky jsou mazány vždy 29. den v měsíci a jsou smazány statistiky starší jednoho měsíce.

Časy tykající se této problematiky jsou napevno zadefinované v souborech pro mazání statistik (*statsDelete.py*, *stats5Delete.py*, *stats15Delete.py*) a v souboru pro periodické spouštění pomocí Cronu. (*stats*).

6.5.4 Cron

Cron v systému Unix umožňuje periodické spouštění skriptů. V adresáři */etc/cron.d* je vytvořen soubor, ve kterém je definováno, jak často má určitý skript spouštět, s jakými právy a také jaká je absolutní cesta k souboru, který má být vykonáván.

K aplikaci byl proto vytvořen soubor *stats*, který definuje opakované spouštění šesti skriptů, které byli popsány již dříve. Nyní následuje výpis obsahu souboru *stats*:

```
# /etc/cron.d/stats: crontab fragment for stats
* * * * *     root   /var/www/SDSKSI/python/byCron/stats.py
0,5,10,15,20,25,30,35,40,45,50,55 * * * *    root
/var/www/SDSKSI/python/byCron/stats5.py
0,15,30,45 * * * *     root   /var/www/SDSKSI/python/byCron/stats15.py
* * * * 0     root   /var/www/SDSKSI/python/byCron/statsDelete.py
* * * * 0    root   /var/www/SDSKSI/python/byCron/stats5Delete.py
* * 29 * *     root   /var/www/SDSKSI/python/byCron/stats15Delete.py
```

Zde je vidno, že například soubor *stats.py* je spuštěn každou minutu, s právy *root* a je umístěn v adresáři */var/www/SDSKSI/python/byCron*.

6.6 Export statistických dat

Statistická data lze exportovat. Odkaz na uložení vygenerovaného souboru je možné nalézt pod tabulkou informující o zobrazeném systému. Soubor je generován s názvem ReceivedData a v závorce s uvedenou IP adresou zařízení. V souboru jsou uloženy všechna statistická data o zařízení, jeho rozhraních, adresní tabulka, směrovací tabulka a net to media tabulka.

Dalším možným exportem je export dat o síťovém provozu na konkretním rozhraní síťového prvku. Soubor je pak ukládán pod názvem throughput, počtem hodin statistiky, IP adresy zařízení a číslem rozhraní, kterého se statistická data o propustnosti týkají.

Jako delimiter mezi různými variable bindings je použit *středník*. Je možné vypozorovat, že pro další rozdělení variable binding na OID a hodnotu lze použít jako delimiter *dvojtečku* a *mezeru*.

7 Instalace potřebného software a aplikace SDSKSI

Dohlížecí stanici je vhodné umístit do podsítě, která je na první úrovni vytvářené topologie. Je ji možné vytvořit i jako virtuální počítač s nainstalovaným OS Debian GNU/Linux spuštěného na jednom z počítačů v síti.

Pro virtualizaci počítače je možné použít program VMware. Po nainstalování OS Debian verze 5.0 a vyšší je zapotřebí nainstalovat několik dalších balíčků do systému. Jednotlivé balíčky jsou blíže popsány spolu s jejich instalací. Na přiloženém CD v příloze je image virtuálního počítače s OS Debian hotov a připraven k použití.

7.1 SSH

Pro vzdálený přístup k virtuálnímu počítači pomocí programu PuTTY a WinSCP je potřeba nainstalovat balík *ssh* příkazem:

```
apt-get install ssh
```

Následující kroky je pohodlnější vykonávat přes vzdálený přístup pomocí SSH a programu PuTTY, běžně dostupného ke stažení z Internetu. K čemu bude použit WinSCP je uvedeno dále.

7.2 Server Apache a PHP

Pro nainstalování PHP do systému je potřeba nainstalovat balíčky *apache2*, *php5* a knihovny *libapache2-mod-php5*, to se provede zadáním příkazu:

```
apt-get install apache2 php5 libapache2-mod-php5
```

Tímto se vytvořil Apache Server, který bude schopný zpracovávat PHP dotazy z webové aplikace. Rovněž se vytvořila složka */var/www*, do které je potřeba nakopírovat celou aplikaci SDSKSI. Nakopírování z hostitelského PC (to je ten, na kterém běží VMware) je možné provést například pomocí aplikace WinSCP, která je taktéž volně dostupná na Internetu. V adresářích aplikace SDSKSI je již umístěna PHP knihovna JpGraph pro generování grafů o síťovém provozu.

7.3 Balíčky pro podporu protokolu SNMP

Aby bylo možné provádět SNMP dotazy, je nutné aby byl pro jazyk PHP nainstalován balíček *php5-snmp* a pro Python balíček *python-pysnmp2*

```
apt-get install snmp snmpd php5-snmp python-pysnmp2
```

7.4 Databáze MySQL

Dále je nutné nainstalovat databázi MySQL a její implementaci do jazyka PHP a Python. To se provede vykonáním příkazu:

```
apt-get install mysql-server-5.0 php5-mysql python-mysqldb
```

a následně balíček *phpmyadmin* pro přístup k databázi přes webové rozhraní:

```
apt-get install phpmyadmin
```

Je-li vše správně nainstalováno, je potřebná zjistit IP adresu virtuálního počítače pomocí příkazu:

```
ip addr
```

a poté do prohlížeče hostitelského PC zadat zjištěnou IP adresu lomeno phpmyadmin. Při správné instalaci všeho předešlého, by se mělo zobrazit přihlašovací okno do databáze MySQL.

Po zadání loginu a hesla vytvářeného při instalaci balíku *mysql* je vhodné vytvořit nového uživatele databáze a novou databázi. Například s názvem SDSKSI.

Do této databáze je pak potřeba nahrát tabulky, jejichž strukturu je možné vytvořit importováním sql skriptu *taps.sql* z adresáře *SDSKSI/zahrnout/*. Tím je databáze připravena a zbývá nainstalovat balík *graphviz*, který slouží pro kreslení neorientovaného grafu topologie sítě.

7.5 GraphViz

Program GraphViz se nainstaluje stejným příkazem pro instalaci jako všechny ostatní balíčky:

```
apt-get install graphviz
```

7.6 Cron Job

Nakonec je ještě nutné zkopírovat soubor *stats* z adresáře */var/www/SDSKSI/python /byCron* do adresáře */etc/cron.d* a provést restart programu Cron:

```
service cron restart
```

Zvláště zde je potřeba dát pozor na to, aby konce řádku souboru *stats* byly správně zalomeny. To lze zkontrolovat příkazem:

```
mcedit /etc/cron.d/stats
```

Soubor nesmí obsahovat znaky *^M* na konci řádků.

7.7 Práva a vlastnictví adresářů a souborů

Pro umožnění zápisu webové aplikace do adresářů je potřeba změnit vlastníka adresářů */var/www/SDSKSI* na vlastníka *www-data* pomocí příkazu:

```
chown -R www-data:www-data /var/www/SDSKSI
```

a nastavit práva *0754* souborům vykonávaným programem Cron v adresáři */var/www/SDSKSI /python/byCron*

```
chmod -R 0754 /var/www/SDSKSI/python/byCron
```

Nyní by měla být aplikace připravena k použití a po správné instalaci všech balíčku by měla být dostupná z webového rozhraní pod IP adresou virtuálního počítače lomeno SDSKSI (velkým písmem).

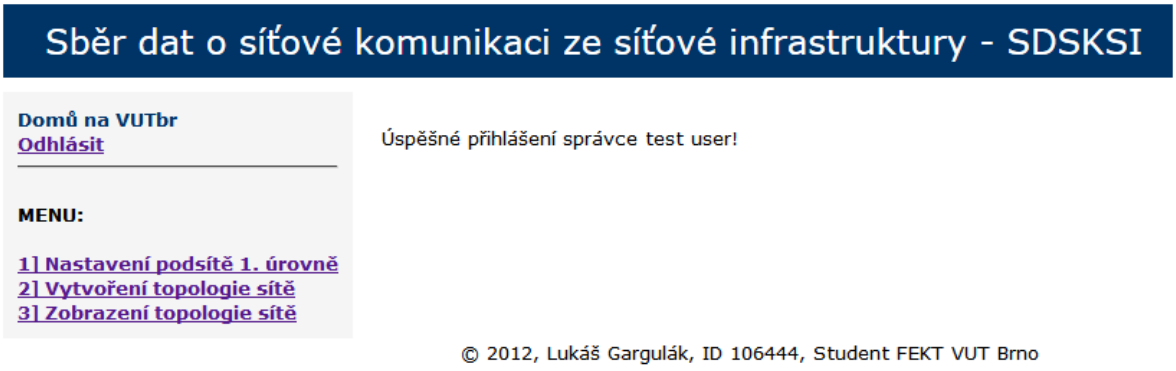
8 Popis webového rozhraní aplikace

Pro přístup do webové aplikace je potřeba se nejprve přihlásit. Výchozí login a heslo je *test*. Další uživatele je možné doplnit SQL příkazem INSERT dle požadavků v literatuře [1]:

```
INSERT INTO admins
(admin_jmeno, admin_login, admin_heslo, admin_date_reg)
VALUES
('Jmeno Prijmeni','login',md5('heslo'), now());
```

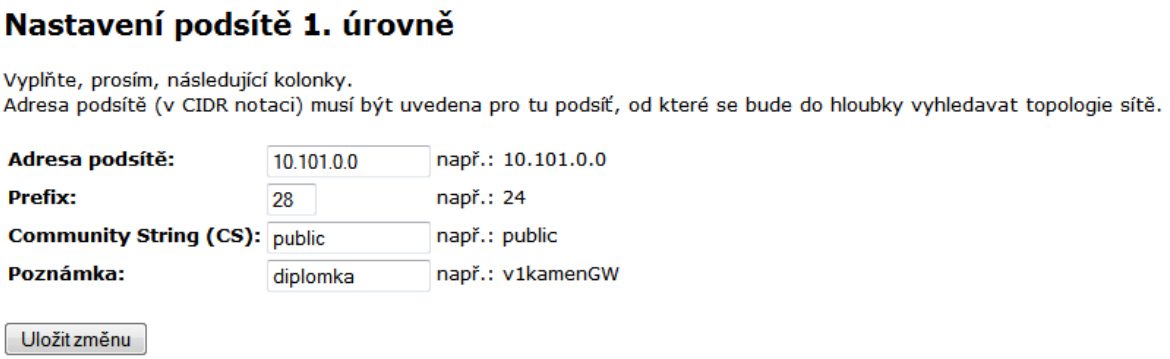
přes rozhraní *phpmyadmin*.

V levé části obrazovky se nachází krátké menu s několika odkazy. Viz obr. 8.1.



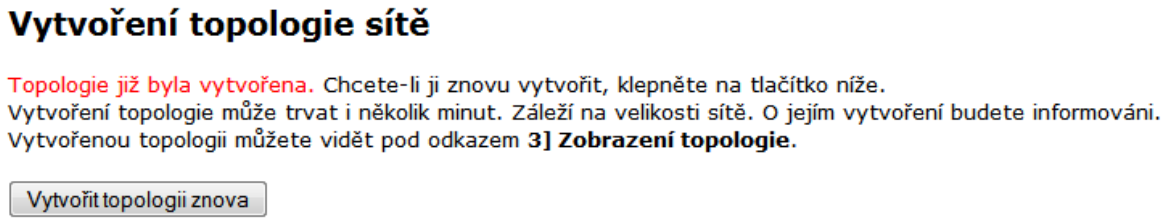
Obr. 8.1: Snímek vzhledu aplikace po přihlášení uživatele s menu na levé straně

Prvním důležitým odkazem je odkaz pro *nastavení podsítě 1. úrovně*. Po kliknutí na něj je potřeba zadat IP adresu podsítě první úrovně a její prefix. Důležitou kolonkou k vyplnění je community string. Poznámka k podsíti není povinná. Obsah toho odkazu znázorňuje obr. 8.2.



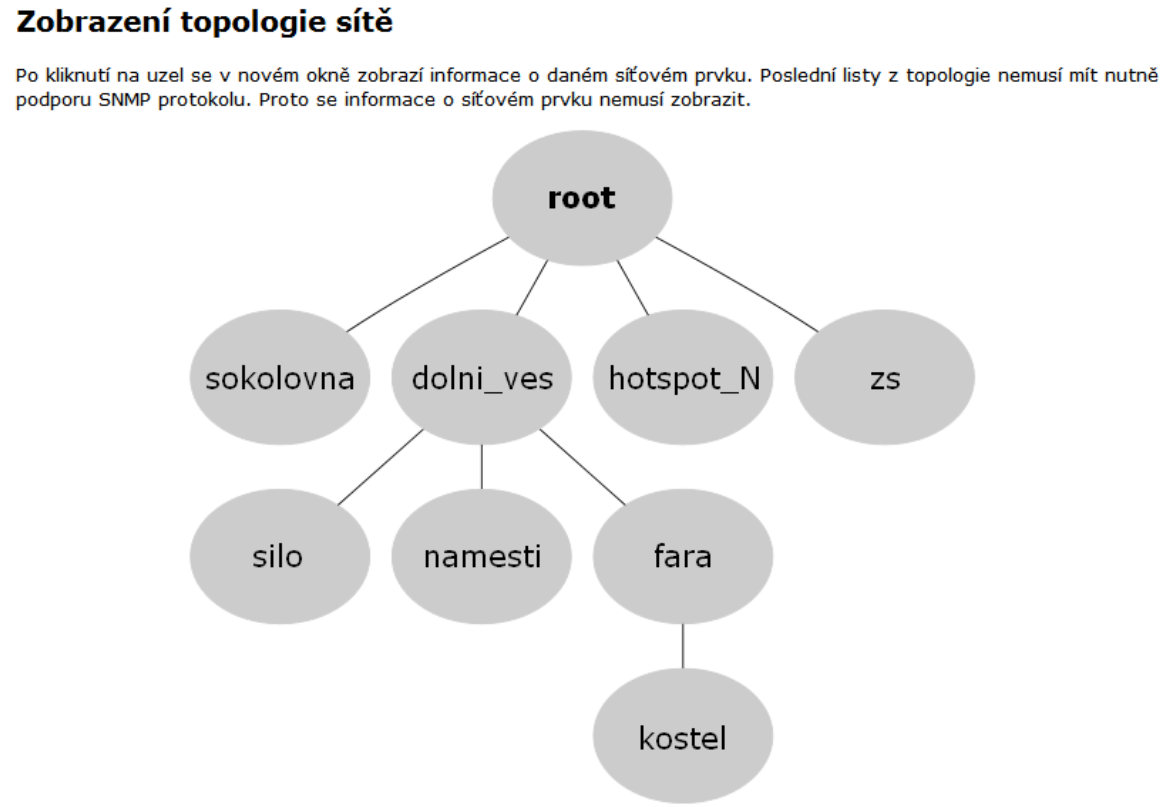
Obr. 8.2: Nastavení podsítě první úrovně

Dalším odkazem je *vytvoření topologie sítě*. Zde je možné topologii sítě vytvořit, nebyla-li ještě vytvořena. Topologii sítě lze vytvořit znova, jestliže byl změněn například prefix adresy podsítě první úrovně. Proces vytvoření podsítě může trvat i několik minut - záleží na velikosti sítě. Obsah tohoto odkazu je zobrazen na obr. 8.3.



Obr. 8.3: Vytvoření topologie sítě

Posledním důležitým odkazem menu je odkaz *zobrazení topologie sítě*. Po kliknutí na tento odkaz se zobrazí obrázek topologie sítě jako je např. obr. 8.4. Obrázek je vektorový ve formátu **.svg*, takže je možné jej přes ctrl a kolečko myši zvětšit, pokud je to nutné.



Obr. 8.4: Topologie sítě

Na každý uzel, představující směrovač v síti, lze kliknout a poté se v novém okně zobrazí statistická data (obr 8.5) o daném zařízení, které jsou popisovány v kap. 6.4.

Informace o terminálu [redacted] s IP adresou 10.101.0.12

Unikátní ID:	4
Popis terminálu:	RouterOS RB532
Privátní OID:	SNMPv2-SMI::enterprises.14988.1
Uptime:	(86631400) 10 days, 0:38:34.00
Kontaktní informace:	N/A
Název terminálu:	hornanka
Umístění terminálu:	N/A
Typ poskytující služby:	78
Počet rozhraní:	6

Stáhnout CSV soubor se statistickými daty o zařízení.

Rozhraní terminálu

Index	Popis	Typ	Max. velikost datagramu	Rychlost	MAC adresa	Admin status	Stav rozhraní	Propustnost
1	ether1	ethernetCsmacd(6)	1500	100000000	0:c:42:4:54:d7	up(1)	up(1)	1H 1D 1W
2	ether2	ethernetCsmacd(6)	1500	100000000	0:c:42:4:54:d8	up(1)	up(1)	1H 1D 1W

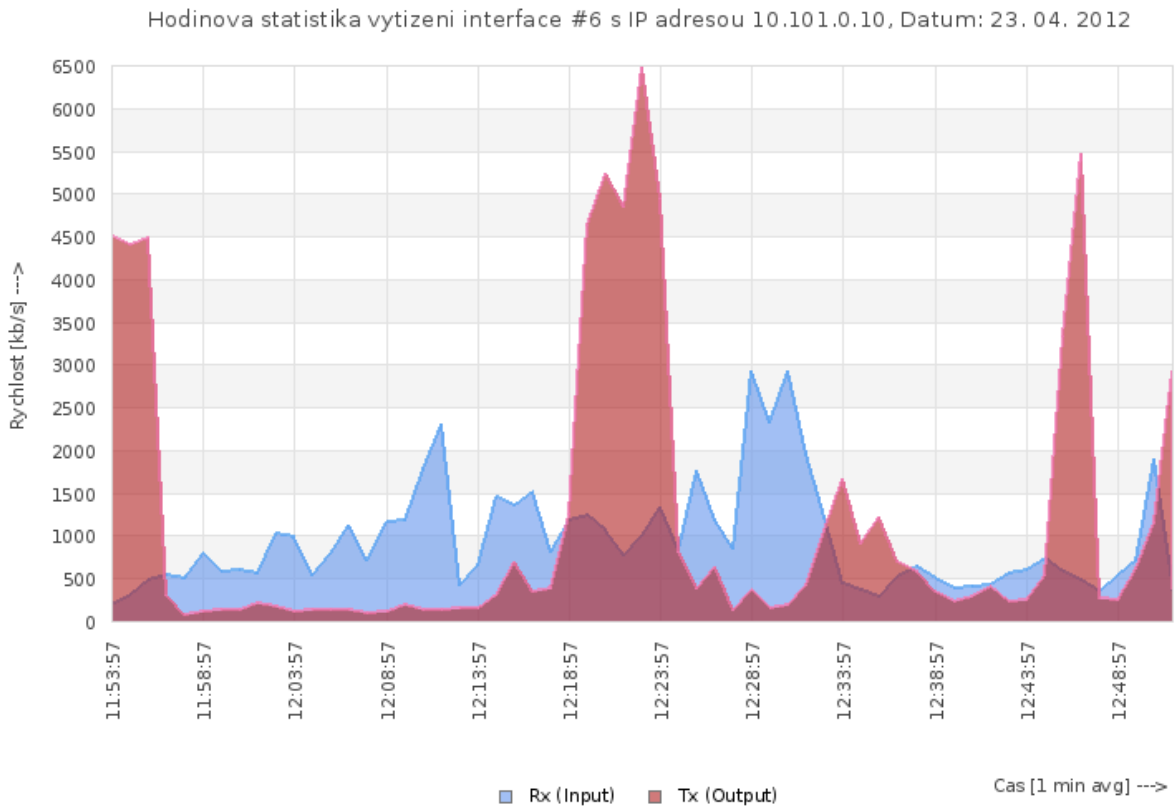
Obr. 8.5: Statistická data o konkretním zařízení v síti s podporou SNMP protokolu

Jak také ukazuje obrázek obr. 8.5 lze tyto statistická data exportovat do souboru.

V tabulce rozhraní terminálu je možné taktéž kliknout na odkazy *1H*, *1D* a *1W*. Tyto tlačítka reprezentují následné zobrazení grafů propustností na jednotlivých rozhraních zařízení za posledních 60 minut (tlačítko *1H*), poslední den (tlačítko *1D*) a poslední týden (tlačítko *1W*). Obr. 8.6 ukazuje graf propustnosti za posledních 60 minut směrovače s IP adresou *10.101.0.10* a na rozhraní *6*.

Po kliknutí na tlačítko *zpět* se dostaneme zpět na aktuální statistická data daného zařízení.

Levý sloupec webové aplikace obsahuje také odkaz na webové stránky VUT v Brně a možnost odhlásit se z aplikace. I po odhlášení sběr dat o síťové komunikaci ze zařízení síťové infrastruktury stále probíhá.



Stáhnout CSV soubor se statistikou propustnosti za posledních 60 minut.

Zpět

Obr. 8.6: Graf propustnosti za posledních 60 minut směrovače s IP adresou 10.101.0.10

9 Laboratorní úloha

Součástí diplomové práce je laboratorní úloha. Tato úloha může být využitelná v předmětech vyučovaných na UTKO FEKT VUT v Brně. Je koncipována jako běžné laboratorní úlohy. Úloha studenty teoreticky i prakticky seznámí s protokolem SNMP.

Po seznámení s teorií bude mít student za úkol nakonfigurovat směrovač Mikrotik a připojit na jeho dvě rozhraní dva počítače s virtualizovanými počítači s OS Debian/GNU Linux. IP adresa bude přidělována DHCP serverem, který studenti taktéž nakonfigurují ve směrovači.

V úloze mají za úkol zjistit hodnoty pro určitá OID z MIB databáze podle [5]. Např. popis zařízení (*sysDescr*) a dobu, po kterou je síťový prvek napájen bez přerušení (*sysUpTime*). Zkusí si prakticky ověřit jaký je rozdíl mezi operacemi SNMP *get* a *get-next*. Poté z jednoho PC budou odesílat data na druhý PC pomocí programu SCP (Secure CoPy) a na základě tohoto přenosu se studenti pokusí vypočítat, kolik oktetů bylo přes rozhraní přeneseno (*ifInOctets* a *ifOutOctets*).

Naposled mají studenti za úkol vytvořit skripty pro směrovač Mikrotik a tyto skripty pomocí protokolu SNMP spouštět. Vyzkouší si tím SNMP operaci *set*.

Celé znění laboratorní úlohy se nachází v příloze A této diplomové práce.

10 Závěr

Pro sběr statistických dat ze zařízení síťové infrastruktury byl vybrán protokol SNMP, protože je nejrozšířenějším protokolem mezi síťovými prvky pro tento účel. Z toho důvodu je rozsáhlá část teoretické části diplomové práce věnována právě detailnímu popisu tohoto protokolu.

Praktická část diplomové práce se zabývá realizací aplikace pro získávání statistických dat ze zařízení síťové infrastruktury a vytvořením topologie sítě se síťovými prvky podporující protokol SNMP . K vytvoření topologie sítě je použito protokolu ICMP a SNMP.

Nejčastěji je vyžadováno na síťových prvcích sledovat jejich využití - rychlost přenosu, počet odeslaných i přijatých paketů, propustnost sítě. Taktéž je možné ze sledovaných prvků zjistit detailní informace jako je typ prvku, výrobce, verzi firmware či OS, umístění prvku, odpovědnou osobu za prvek apod. Sledováním statistických dat ve vytvořené aplikaci také můžeme předcházet problémům v síti. Například vysledujeme-li zvyšující se počet zahazovaných nebo chybných paketů.

Všechny tyto informace je možné pomocí vytvořené aplikace SDSKSI sledovat v přehledném webovém rozhraní. Toto webové rozhraní je naprogramováno v jazyce PHP. Pro grafické doladění je použito jazyka XHTML (eXtensible HyperText Markup Language) s kaskádovými styly CSS (Cascading Style Sheets). Vlastní sběr statistik je prováděn pomocí skriptovacího jazyka Python a knihovnou PySNMP.

Správce sítě si po přihlášení do aplikace může vybrat, ze kterého zařízení chce vypsat statistiky. Tyto zařízení jsou zobrazeny ve stromové struktuře, která vyobrazuje správci představu o topologii sítě. Topologii je možné vytvářet několikrát a to vždy na zvoleném rozsahu IP adres první úrovně. Statistická data původních zařízení v síti nejsou ztracena. Avšak jsou mazány statistiky, které jsou starší a nejsou považována za důležitá. Tuto činnost ošetřuje Cron v OS Debian.

Aplikace by mohla obsahovat spoustu rozšíření. Jak již bylo uvedeno v textu práce, pomocí SNMP protokolu je také možné měnit některé parametry síťového prvku. Avšak není jich mnoho. Nejčastěji se využívá vypnutí nebo zapnutí jednotlivých portů, nebo omezení propustnosti na jednotlivých portech. Tyto možnosti by mohly být vhodným rozšířením aplikace SDSKSI. K řešení by se využívala enterprise MIB od různých výrobců daného zařízení.

Operace SNMP set není v aplikaci využívána. Nicméně v laboratorní úloze se s ní studenti setkají a funkci této SNMP operace si prakticky ověří.

Dalším vylepšením aplikace by mohlo být, že vyobrazená topologie sítě by byla zpracována javascriptem a za pomoci protokolu ICMP by mohla zobrazovat správci sítě, zda je zařízení aktuálně dostupné nebo nedostupné. Např. zelená barva by znázorňovala dostupnost prvku a červená jeho nedostupnost v síti.

Za poměrně velkou změnu, která by mohla být implementována, by byla podpora protokolu IPv6. Vytvořená aplikace je funkční pouze na síti s protokolem IPv4.

Při dalším vývoji aplikace by bylo vhodné se zaměřit na její rychlost a zdrojový kód co nejvíce optimalizovat pro svůj běh.

Poskytované statistiky by bylo vhodné doplnit o statistická data, které by informovala o vytížení sítě SNMP provozem.

Literatura

[1] LACKO, Luboslav: SQL, kapesní přehled.
Computer Press, a.s., 2005. 96 s. ISBN 80-251-0788-4.

[2] LAMMLE, Tod: CCNA vyukový průvodce přípravou na zkoušku 640-802.
Computer Press, a.s., 2010. 928 s. ISBN 978-80-251-2359-1.

[3] MAURO, Douglas; SCHMIDT, Kevin: Essential SNMP. Second Edition.
O'Reilly Media, 2005. 464 s. ISBN 978-0-596-00840-6.

[4] ULLMAN, Larry: PHP a MySQL.
Computer Press, a.s., 2004. 534 s. ISBN 80-251-0063-4.

[5] ALVESTRAND, H. Tveit: Object Identifiers [online].
www.alvestrand.no - [cit. 15.10. 2011].
Dostupné na www: <http://www.alvestrand.no/objectid>.

[6] ASIAL CORPORATION: Most powerful PHP-driven charts [online].
JpGraph - [cit. 18.3. 2012].
Dostupné na www: <http://jpgraph.net/>.

[7] CISCO SYSTEMS.: Introduction to Cisco IOS NetFlow - A Technical Overview [online].
Cisco Systems, Inc - [cit. 10.11. 2011].
Dostupné na www:
<http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/prod_white_paper0900aecd80406232.html>.

[8] ELKNER, Jeffrey: How to think like a computer scientist-Learning with Python [online].
howto.py.cz - [cit. 14.1. 2012].
Dostupné na www: <http://howto.py.cz/index.htm>.

[9 ETINGOF, Ilya: PySNMP tutorial [online].
pysnmp.sourceforge.net - [cit. 4.2. 2012].
Dostupné na www: <http://pysnmp.sourceforge.net/docs/4.x/index.html>.

[10] GANSNER, Emden: Layout Commands: DOT [online].
AT&T Research - [cit. 7.3. 2012].
Dostupné na www: <http://www.graphviz.org/pdf/dot.1.pdf>.

[11] MIKROTIK: Manual:SNMP [online].
www.mikrotik.com - [cit. 24.4. 2012].
Dostupné na www: <http://wiki.mikrotik.com/wiki/SNMP>.

[12] NAGIOS ENTERPRISES: Nagios [online].
Nagios Enterprises, LLC - [cit. 4.11. 2011].
Dostupné na www: <http://nagios.org/>.

[13] NET-SNMP: Tutorials [online].
www.net-snmp.org - [cit. 4.2. 2012].
Dostupné na www: <http://www.net-snmp.org/wiki/index.php/Tutorials>.

[14] OETIKER, Tobi: The Multi Router Traffic Grapher [online].
MRTG - [cit. 4.11. 2011].
Dostupné na www: <http://oss.oetiker.ch/mrtg/>.

[15] SAMURAJ: Zařízení v síti pod kontrolou [online].
Samuraj-cz.com - [cit. 16.10. 2011].
Dostupné na www: <http://www.samuraj-cz.com/clanek/zarizeni-v-siti-pod-kontrolou/>.

[16] THE CACTI GROUP: What is Cacti? [online].
The Cacti Group, Inc - [cit. 4.11. 2011].
Dostupné na www: <http://www.cacti.net/what_is_cacti.php>.

[17] TUTORIALSPOINT: Python - MySQL Database Access [online].
www.tutorialspoint.com - [cit. 22.2. 2012].
Dostupné na www: <http://www.tutorialspoint.com/python/python_database_access.htm>.

Seznam zkratek

1D – One day

1H – One hour

1W – One week

ASN.1 – Abstract Syntax Notation One

CCITT – Comité Consultatif International Téléphonique et Télégraphique

CD – Compact disc

CRC – Cyclic Redundancy Check

CSS – Cascading Style Sheets

DES – Data Encryption Standard

DoD – Department of Defence

DVD – Digital Versatile Disc

EGP – Exterior Gateway Protocol

FTP – File Transfer Protocol

GPL – General Public License

ICMP – Internet Control Message Protocol

IP – Internet Protocol

ISO – International Organization for Standardization

MGMT – ManaGeMenT

MIB – Management Information Base

MRTG – Multi Router Traffic Grapher

NAT – Network Address Translation

NMS – Network Management System

OID – Object Identifier

OS – Operation System

PC – Personal computer

PDU – Packet Data Unit

PHP – Hypertext Pre-Processor

PING – Packet InterNet Groper

RFC – Request for Comments

RRD – Round-Robin Database

SDSKSI – Sběr statistických dat o síťové komunikaci ze zařízení síťové infrastruktury

SGMP – Simple Gateway Monitoring Protokol

SMI – Structure of Management Information

SMS – Short Message Service

SNMP – Simple Network Management Protocol

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

VoIP – Voice over Internet Protocol

VPN – Virtual Private Network

XHTML – eXtensible HyperText Markup Language

A Laboratorní úloha

Seznámení se s protokolem SNMP

Cíl

Seznámení se s protokolem SNMP a praktické získavání dat pomocí balíku NET-SNMP v OS Debian GNU/Linux.

Požadavky na pracoviště

Směrovač Mikrotik RouterBoard 433AH + napájecí zdroj,

2x PC,

2x UTP kabel,

VMware,

virtuální PC s OS Debian 5.0 GNU/Linux.

Úkoly

1. Seznámit se s protokolem SNMP, strukturou MIB databáze, příkazy NET-SNMP pro vykonávání SNMP operací.
2. Zapojit síťové prvky na pracovišti a nakonfigurovat směrovač.
3. Ověřit logické propojení virtuálních PC přes směrovač.
4. Zjistit hodnoty určitých OID MIB databáze.
5. Přenést soubor mezi dvěma počítači a pomocí SNMP ověřit jeho velikost.
6. Vyzkoušení si SNMP operace *set* - vypnutí rozhraní směrovače
7. Psaní skriptu pro směrovač Mikrotik

Teoretický úvod

SNMP - Simple Network Management Protocol

Pro sběr statistických dat o síťové komunikaci je možné využít Simple Network Management Protocol, dále jen SNMP. Tyto data získává protokol SNMP ze sledovaného prvku pomocí SNMP agenta.

Historie protokolu SNMP a jeho verze

Protokol SNMP se vyvinul z protokolu SGMP (Simple Gateway Monitoring Protokol). Začal se využívat v roce 1988 a brzy se stal nejrozšířenějším protokolem pro řízení a sledování počítačových sítí. Podpora SNMP protokolu je v dnešní době rozšířena nejen u serverů a aktivních síťových prvků, ale téměř u všech zařízení, které komunikují po počítačové síti (tiskárny, záložní zdroje, bezdrátové přístupové body, či různá síťová čidla, jako například teploměr apod.).

První verze tohoto protokolu, SNMPv1, je specifikována v RFC 1157. Obsahuje řadu nedostatků. Například autentizace pomocí hesla, tzv. *community string*, je nedostatečná. Toto heslo se přes sít nepřenáší v šifrované formě.

Verze protokolu SNMPv2 existuje v několika variantách. Tento standard začal vznikat v roce 1993 a později, v roce 1996, byl upraven. Nejrozšířenějším standardem mezi výrobci síťových zařízení je dnes standard ve variantě SNMPv2c popisovaný v RFC 1901 až RFC 1908.

Z důvodu nedostatečné bezpečnosti obou předchozích verzí vznikla na jaře roku 1998 ještě novější verze - SNMPv3. Předchozí verze měly ochranu pro přenos dat pouze pomocí community string. V této verzi je již komunikace šifrována DES algoritmem. V současné době tento standard protokolu SNMP není tolik rozšířen mezi výrobci síťových prvků. Standard SNMPv3 je definován v RFC 3411 až RFC 3418.

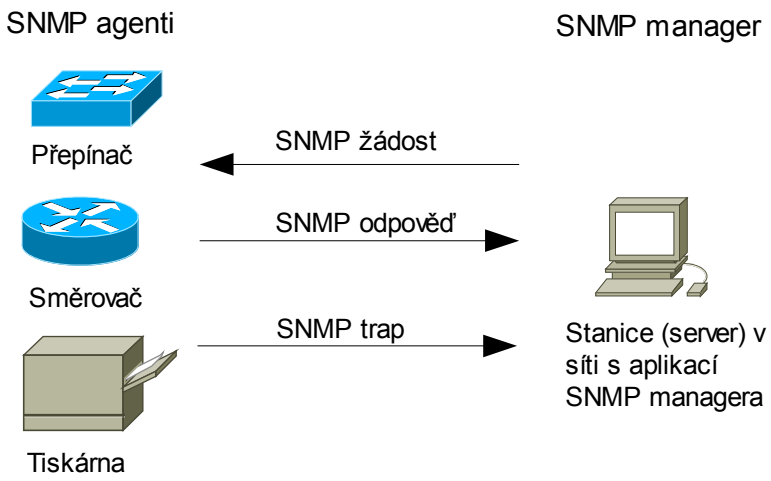
Cíle a princip SNMP

Při vzniku protokolu SNMP byl kladen požadavek vytvořit pro administrátory sítí univerzální protokol pro sledování stavu sítě a jejich vzdálené řízení. Protokol SNMP je založený na modelu *klient* / *server*. Klient je označován jako SNMP manager a server jako SNMP agent.

SNMP manager

SNMP manager je program, který běží na síťové stanici. Ve větších sítích se jako manager často používá NMS (Network Management System), který běží na pracovní stanici vyhrazené pro tento účel. Pomocí NMS pak administrátor sítě může číst nebo také nastavovat jednotlivé parametry sledovaných a řízených prvků v počítačové síti pomocí dotazování SNMP agenta pomocí SNMP operací. Jak SNMP manažer komunikuje s jednotlivými agenty, je naznačeno

na obr. 1. Manager posílá dotazy agentovi a přijímá jeho odpovědi. Hodnoty ze sledovaných prvků může získávat i více SNMP managerů najednou.



Obr. 1: Princip SNMP

SNMP agent

SNMP agent je program, který běží na zařízení připojeném do počítačové sítě a odpovídá na dotazy SNMP managera. Agent neustále monitoruje a sbírá informace o všech dostupných funkcích a stavech daného zařízení. K získání určité informace ze zařízení, na kterém beží SNMP agent, musí SNMP manager vyslat požadavek na dané zařízení a projít informace poskytované agentem.

Mimo jiné SNMP agent může vysílat asynchronně hlášení o poruše (tzv. *trapy*) na adresu definovanou administrátorem. Tyto trapy jsou odesílány bez žádosti SNMP managera. Může se jednat o zasílání různých hlášení o poruše zařízení v počítačové síti, výpadku napájení, výpadku ventilátorů, vysoké teplotě apod.

Bezpečnost SNMP a přístupová práva SNMP managerů k SNMP agentům

Přístup k jednotlivým objektům agenta je důležité nějak zabezpečit. Jedná se vlastně o tzv. přístupové práva SNMP managerů k SNMP agentům. Tohoto zabezpečení se dosáhlo velice primitivním principem. Tím, že každý paket ve své hlavičce nese pole označené *community string*. Toto pole funguje podobně jako kombinace uživatelského jména a hesla.

V praxi je definován jeden *community string* pro čtení i zápis (*read-write*) a jeden *community string* pro omezený přístup, pouze pro čtení (*read-only*). Jestliže je tedy ve sledovaném zařízení uložen stejný *community string*, jako v příkazu od SNMP managera, provede se požadovaná akce podle typu operace. Nebudou-li souhlasit, bude akce odmítnuta.

Informace posílané v SNMP paketech jsou uloženy v čistém textu. To znamená, že i *community string* je přenášen jako čistý text a je možné jej síťovým analyzátorem odchytit.

Proto přišla verze SNMPv3, která přinesla několik vylepšení pro oblast bezpečnosti a komunikaci šifruje pomocí DES algoritmu.

Ve výchozím nastavení je u SNMP agentů nastaven *community string* pro čtení na hodnotu public a pro čtení a zápis na hodnotu private.

Protože *community string* je v podstatě heslo, měly by se při výběru zohledňovat obecné kriteria a pravidla pro vytvoření hesla. Neměla by se používat slovníková slova. Heslo by se mělo skládat z číslic i písmen, velkých i malých.

Je důležité si uvědomit, kdo má přístup pro čtení i zápis k SNMP zařízením a kdo tak může získat kontrolu nad těmito zařízeními použitím protokolu SNMP (např. může nastavovat rozhraní směrovače, přepínat stavy portů nebo měnit směrovací tabulky).

Jednou z cest, jak se chránit před tímto útokem je používat virtuální soukromou síť VPN (Virtual Private Network) a zajistit tak, že provoz je šifrován.

Druhou z cest je častá změna *community string*. Změna *community string* není obtížná v malých sítích. Jedná-li se však o síť se stovkami síťových prvků, ze kterých jsou sbírány statistické údaje, je vhodné pro změnu *community string* napsat skript.

Pro snížení pravděpodobnosti útoku je možné nastavit IP firewall nebo filtry tak, že je v nich např. povolen UDP (User Datagram Protokol) provoz pouze od známých zařízení. Je také možné povolit UDP provoz pouze na port 161 (pro SNMP žádosti) nebo na port 162 (pro trapy). Firewall není stoprocentně efektivní řešení, ale určitě dokáže zredukovat pravděpodobnost útoku na zařízení v síti.

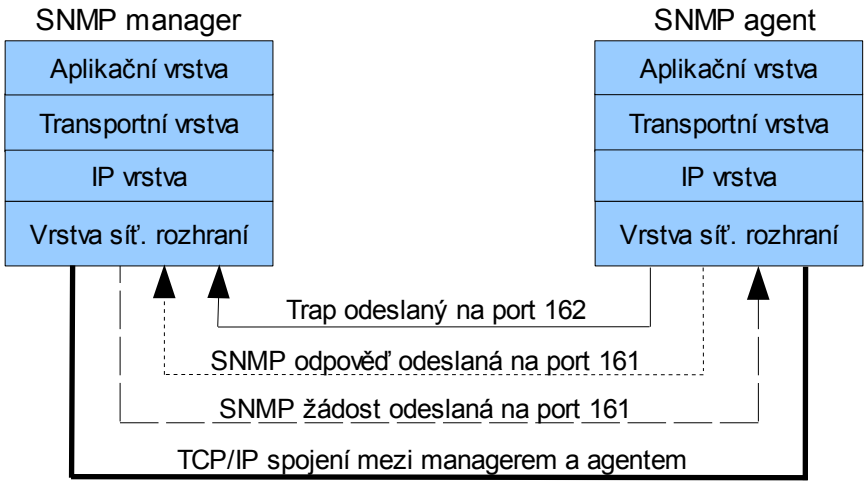
Komunikační protokol SNMP

SNMP pro komunikaci využívá protokol UDP ze sady TCP/IP. UDP bylo vybráno namísto TCP, jelikož u UDP nevytváří spojení mezi managerem a agentem před přenosem požadované zprávy. Protokol UDP je také tzv. nespolehlivým protokolem, jelikož příjemce nepotvrzuje přijetí zprávy od odesílatele. Jestli byla zpráva ztracena nebo přijata, má na starost jednoduchý časový limit (tzv. *timeout*) programu SNMP managera. Manager posílá UDP SNMP požadavky na agenta a čeká na odpověď. Jak dlouho budeme čekat na odpověď záleží na nastaveném časovém limitu v programu SNMP managera. Jestliže doba od odeslání dosáhne časového limitu a manager do této doby neobdržel odpověď na svůj požadavek,

znamená to, že paket byl ztracen. Poté se může požadavek vyslat znova. Kolikrát, to záleží opět na nastavení programu SNMP managera.

Jestliže se jedná o tyto pravidelné a opakovatelné žádosti SNMP managera na odpovědi od SNMP agentů, není nespolehlivost UDP tak závažným problémem. Maximálně SNMP manager neobdrží odpověď na svou žádost. Poněkud horší je to v případě zasílání trapů. Jestliže SNMP agent odešle trap a trap není přijat SNMP managerem, tak SNMP manager ani neví, že mu byl trap vyslán. Zároveň ani agent neví, že má trap poslat znova, protože jej o trapy nikdo nežádá. Jak již bylo dříve popsáno trapy jsou odesílány bez vyžádání SNMP managera.

Když SNMP manažer posílá dotaz SNMP agentovi, tak používá zdrojový a cílový port 161. Při odesílání odpovědi se používá tentýž port. Trapy jsou odesílány na port 162. U všech zařízení podporující SNMP protokol jsou tato čísla portů výchozí.



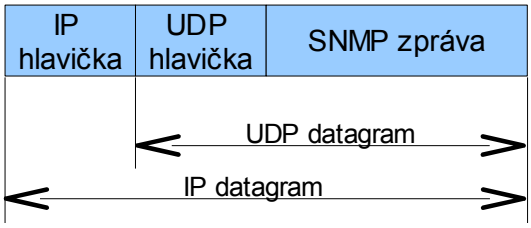
Obr. 2: TCP/IP komunikační model

Na obr. 2 je uveden TCP/IP komunikační model a naznačeno SNMP spojení. Když chce SNMP manager zažádat o nějakou hodnotu od SNMP agenta, nebo když SNMP agent chce odeslat trap, tak na jednotlivých vrstvách vykonává následující:

- Aplikační vrstva – První se program SNMP managera (nebo SNMP agenta) rozhodne, jakou činnost provádět. Například může zaslat SNMP žádost na agenta (nebo agent poslat odpověď na SNMP žádost, či trap).
- Transportní vrstva – Transportní vrstva umožňuje komunikovat SNMP manageru a SNMP agentovi pomocí UDP protokolu. UDP hlavička paketu obsahuje cílový port zařízení, kam je zasílána SNMP žádost nebo trap.
- IP vrstva – Tato vrstva má za úkol doručit SNMP paket příjemci, podle jeho IP adresy.

- Vrstva síťového rozhraní – Poslední vrstva, která musí SNMP paket doručit na místo určení (např. od agenta k managerovi).

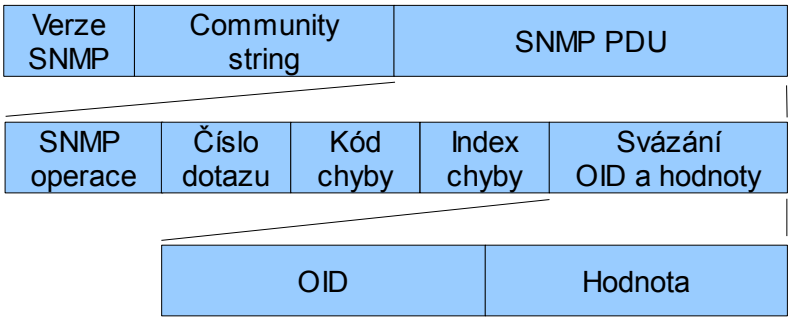
Každá SNMP zpráva je před samotným přenosem zapouzdřena do IP paketu. Tento paket je znázorněn na obr. 3.



Obr. 3: SNMP zpráva zapouzdřená do IP paketu

Formát SNMP paketu

SNMP paket je složen ze dvou částí - hlavičky paketu a vlastních uživatelských dat. Ta jsou označována jako PDU (Protocol Data Unit), viz obr. 4. Hlavička paketu obsahuje číslo verze protokolu a community string.



Obr. 4: SNMP paket

Vlastní datová část PDU je složena z:

- SNMP operace - jedná se o typ operace (jsou popsány dále),
- čísla dotazu - pro svázání dotazu s odpovědí,
- chybového kódu - indikuje, kde k chybě došlo a určuje její typ,
- ukazatele na objekt - přiřazuje chybu dané proměnné z pole pro svázání identifikátoru objektu (OID – Object Identifier) a hodnoty,
- svázání OID a hodnoty - přiřazuje daným proměnným jejich aktuální hodnoty (vlastní data PDU).

MIB databáze

Spravované objekty, které je možné přes protokol SNMP číst nebo nastavit, jsou uloženy v databázi. Této databázi se říká Management Information Base. MIB má hierarchickou stromovou strukturu. Na kořen jsou připojeny uzly. Tyto uzly mohou být dále kořeny jednotlivých podstromů. Každý uzel má své jméno. Toto jméno je složeno jak z textového řetězce, tak také z čísla.

MIB databáze je tvořena z několika modulů (tříd). Jednotlivé moduly obsahují vždy skupinu souvisejících údajů.

Tyto moduly jsou uloženy jako textové soubory, které mají přesně definovanou strukturu, aby SNMP manager byl schopný informace o sledovaném zařízení v síti získat a SNMP agent tyto informace předat. Používá se notace SMI (Structure of Management Information), která je součástí ASN.1 (Abstract Syntax Notation One). S příchodem protokolu verze SNMPv2 se základní notace SMIv1 (SMI verze 1) rozšířila o několik možných spravovaných objektů na verzi SMIv2.

Přístup k objektu v MIB databázi

Pro přístup k objektu jsou používány celá čísla, která jsou oddělena tečkou. Této sekvenci čísel se říká Object Identifier – OID. Na obr. 5 je uvedena část struktury MIB databáze dle RFC 1213.

Např. pro zjištění popisu sledovaného zařízení bude OID v číselném vyjádření:

```
.1.3.6.1.2.1.1.1
```

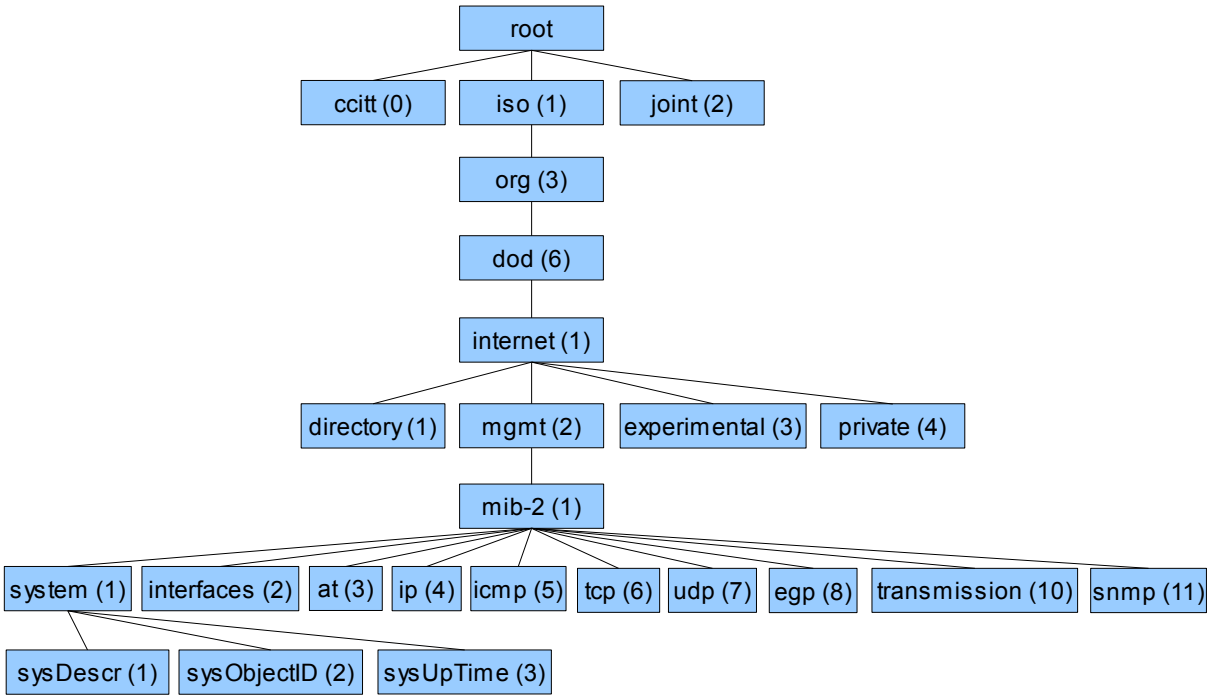
Dotaz SNMP agentu je možné poslat i ve formě textového řetězce. Pak by dotaz zněl:

```
.iso.org.dod.internet.mgmt.mib-2.system.sysDescr
```

Nyní bude podle obr. 5 popsáno, co definují jednotlivé podstromy:

- system – 1.3.6.1.2.1.1 – Definuje seznam objektů, které se týkají fungování systému jako je např. doba běhu systému, kontakt na správce zařízení, jméno zařízení, apod.
- interfaces – 1.3.6.1.2.1.2 – Uchovává informace o stavu jednotlivých rozhraních zařízení. Jestli je rozhraní ve vypnutém (anglicky down) nebo zapnutém stavu (anglicky up). Umožňuje také sledovat, kolik oktetů bylo přeneseno přes jednotlivá rozhraní v obou směrech, apod.
- at – 1.3.6.1.2.1.3 – Je podstrom pro překlad adres. Již se nepoužívá.

- ip – 1.3.6.1.2.1.4 – Uchovává mnoho záznamů souvisejících s IP protokolem. Zahrnuje taktéž směrovací tabulky.
- icmp – 1.3.6.1.2.1.5 – Obsahuje záznamy o chybách ICMP protokolu.
- tcp – 1.3.6.1.2.1.6 – Obsahuje záznamy o stavu TCP spojení.
- udp – 1.3.6.1.2.1.7 – Obsahuje statistiky UDP provozu.
- egp – 1.3.6.1.2.1.8 – Obsahuje záznamy o statistikách protokolu EGP.
- transmission – 1.3.6.1.2.1.10 – Zatím neobsahuje žádné definice.
- snmp – 1.3.6.1.2.1.11 – Zaznamenává statistiky o provozu SNMP. Umožňuje například sledovat počet odeslaných a přijatých SNMP paketů.



Obr. 5: Část struktury MIB databáze dle RFC 1213

Popis operací protokolu SNMP balíku Net-SNMP

V předcházejícím textu bylo již několikrát zmíněno, že existují operace SNMP. Mezi tyto operace patří:

- *get*
- *get-next*
- *get-bulk* (pouze v SNMPv2 a SNMPv3)

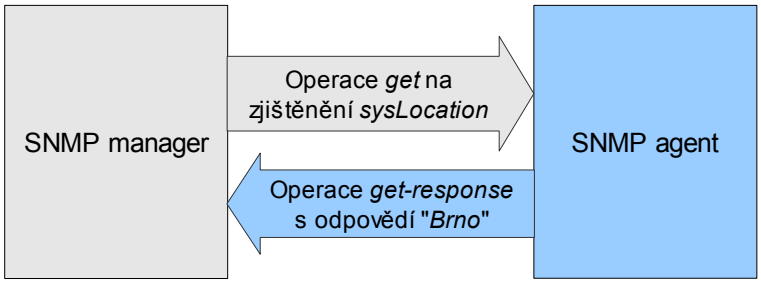
- *set*
- *get-response*
- *trap*
- *notification* (pouze v SNMPv2 a SNMPv3)
- *inform* (pouze v SNMPv2 a SNMPv3)
- *report* (pouze v SNMPv2 a SNMPv3)

V následujících odstavcích budou popsány pouze operace potřebné pro tuto laboratorní úlohu.

Operace get

Operace *get* je operace zahajovaná SNMP managerem. Jedná se o požadavek na SNMP agenta. Agent tento požadavek zpracovává a posílá zpět odpověď k SNMP managerovi. Může však nastat situace, kdy je požadavek na straně SNMP agenta zahozen. To se může stát např. u směrovačů, které jsou provozně velmi vytížené a nezvládají požadavek *get* přijmout. Operace *get* je znázorněna na obr. 6.

Jednou položkou v SNMP paketu je svázání OID a hodnoty. Anglicky je toto pole označováno jako variable bindings. Toto pole je seznam MIB objektů, které dovolují na požádání zjistit, co chce SNMP manager vědět. Příjemce, SNMP agent, pak žádost vyplní (přiřadí hodnotu k určitému OID) a odešle odpověď zpět SNMP managerovi.



Obr. 6: Princip SNMP operace get

Na ukázku je zde uveden konzolový výpis použití operace *get*:

```
# snmpget -v 2c -c public 10.6.1.1 .1.3.6.1.2.1.1.6.0
SNMPv2-MIB::sysLocation.0 = ""
```

Při zadávání operace *get* je důležité si uvědomit, že je zadávána z SNMP managera. Pro vykonání operace *get* byl použit příkaz *snmpget*. Tento příkaz je obsažen v balíčku Net-SNMP pro unixový operační systém. Jako argument příkazu *snmpget* je zadána verze SNMP *2c*, druhý argument je pro community string (veřejný - *public*), třetí argument je IP

adresa zařízení a čtvrtý identifikátor objektu *1.3.6.1.2.1.1.6.0*, na jehož hodnotu se dotazujeme.

Z kapitoly o MIB databázi je známo, že OID *1.3.6.1.2.1.1* je určen pro podstrom *system*. V dotazovaném OID jsou ale uvedeny ještě další dvě celé čísla. *6* a *0*.

6 je aktuální proměnná v MIB, na kterou je dotazováno (*sysLocation*). Jak je vidět z konzolového výpisu, tak odpověď je *system.sysLocation.0 = ""*. To znamená, že *sysLocation* není na směrovači nastaven. Také můžeme vidět, že odpověď je zachována ve formátu variable binding. *OID = hodnota*.

Na konci OID je umístěna nula. V SNMP jsou objekty MIB definovány konvencí *x.y*, kde *x* představuje aktuální OID proměnné (*1.3.6.1.2.1.1.6*) a *y* představuje identifikátor instance. Pro skalární objekty je *y* vždy *0*. Skalární objekty jsou objekty, které nejsou definovány jako řádek v tabulce. Jestliže chceme vybrat u ostatních objektů určitý řádek tabulky, tak místo nuly zapíšeme číslo řádku.

Operace *get* je užitečná pro získávání jednoho MIB objektu v jeden určitý časový okamžik. Jestliže v jeden časový okamžik je požadováno ze spravovaného zařízení získat více objektů, je vhodné použít SNMP operaci *get-next*.

Operace get-next

Operací get-next je možné načíst více hodnot z MIB. Operace get-next prochází podstrom postupně. Jelikož OID je sekvence celých čísel, tak je pro SNMP agenta nejjednodušší začít hledat u kořene stromu a postupně vyhledat potřebný OID. Tato metoda vyhledávání je odborně označována jako vyhledávání do hloubky. Když SNMP manager příjme odpověď od SNMP agenta na požadavek *get-next*, vyšle manager další požadavek *get-next*. Tuto činnost provádí tak dlouho, dokud mu není od SNMP agenta vrácena chyba, že nebylo dosaženo konce MIB a neexistují další objekty, které by byly zaslány s odpovědí.

Postupné zadávání operace get-next umožňuje příkaz snmpwalk. Zde je uvedena ukázka jeho použití:

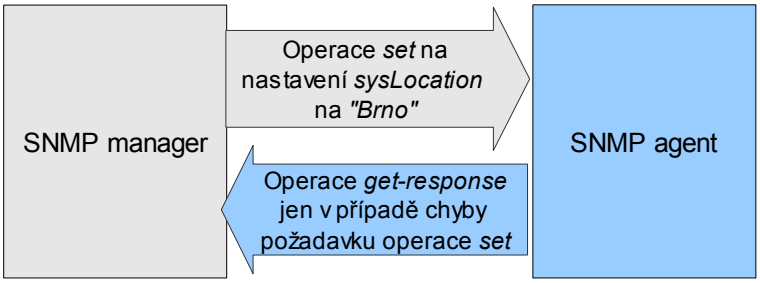
```
# snmpwalk -v 2c -c public 10.6.1.1 .1.3.6.1.2.1.1
SNMPv2-MIB::sysDescr.0 = STRING: RouterOS RB532A
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.14988.1
DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (93551600) 10 days,
19:51:56.00
SNMPv2-MIB::sysContact.0 = STRING:
```

```
SNMPv2-MIB::sysName.0 = STRING: Msafar
SNMPv2-MIB::sysLocation.0 = STRING:
SNMPv2-MIB::sysServices.0 = INTEGER: 78
```

Příkaz snmpwalk vrátil sedm MIB proměnných. Každá proměnná je částí podstromu *system* (dle RFC 1213). Je zde například vidět ID zařízení, jak dlouho je zařízení v provozu, kontaktní osobu apod.

Operace set

Operace *set* se používá pro změnu hodnoty proměnné spravovaného objektu nebo pro vytvoření nového řádku v tabulce. Proměnné jsou definovány v MIB jako *read-write* nebo *write-only* a mohou být změněny nebo vytvořeny pomocí této operace. Pomocí této operace je také možné, aby manager nastavil více objektů najednou v jeden časový okamžik. Jestliže nastavení jednoho objektu selže, tak selže nastavení všech a žádné změny vyvolávané touto operací nebudou uskutečněny.



Obr. 7: Princip SNMP operace set

Obr. 7 ukazuje použití operace *set* a na následujícím výpisu můžeme vidět ukázku jejího použití:

```
# snmpget -v 2c -c public 10.6.1.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = ""
# snmpset -v 2c -c private 10.6.1.1 system.sysLocation.0 s "Brno"
SNMPv2-MIB::sysLocation.0 = "Brno"
# snmpget -v 2c -c public 10.6.1.1 system.sysLocation.0
SNMPv2-MIB::sysLocation.0 = "Brno"
```

Nejprve je pomocí příkazu *snmpget* (operace *get*) vypsáno, jak je *sysLocation* nastaven. Z odpovědi SNMP agenta je vidno, že nastaven není. Poté je zadán příkaz *snmpset* s parametry *s* a *"Brno"* (operace *set*). Parametr *s* říká, že proměnná *sysLocation* bude

nastavena jako řetězec. Nastavovaný řetězec je pak napsán mezi uvozovky. To, že *sys.Location* vyžaduje řetězec je určeno podle definice *sysLocation* v RFC 1213:

```
sysLocation OBJECT-TYPE
    SYNTAX  DisplayString (SIZE (0..255))
    ACCESS  read-write
    STATUS  mandatory
    DESCRIPTION
        "The physical location of this node (e.g., 'telephone closet,
         3rd floor')."
    ::= { system 6 }
```

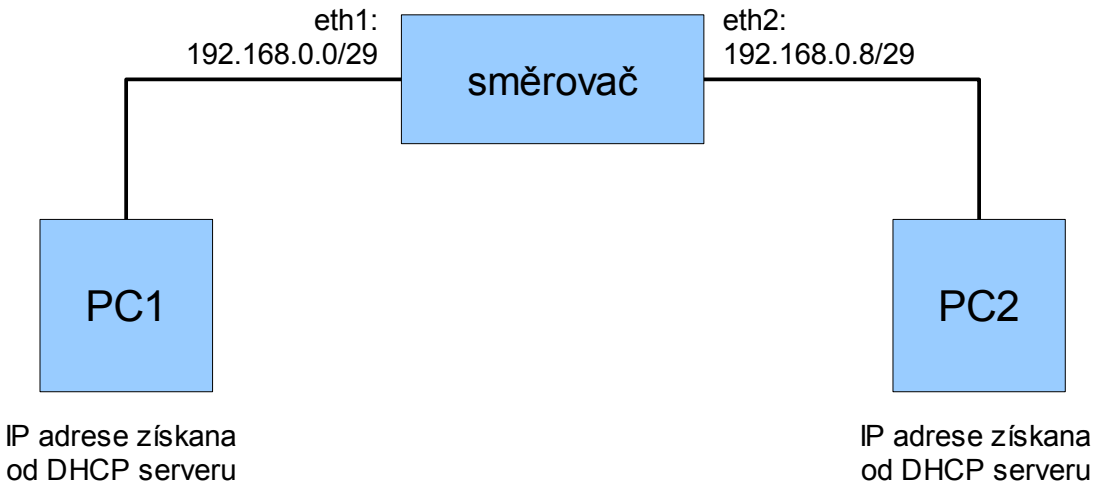
SYNTAX pro *sysLocation* je *DisplayString (SIZE (0..255))*. To znamená, že se může jednat o řetězec s maximální délkou 255 znaků.

Nakonec je v příkladu opět použit příkaz *snmpget*, aby bylo ověřeno, že hodnota proměnné *sysLocation* je opravdu nastavena.

Postup řešení laboratorní úlohy

1) Seznamte se s protokolem SNMP, strukturou MIB databáze a s balíkem NET-SNMP pro práci s SNMP protokolem v prostředí Debian GNU/Linux. Celou strukturu MIB databáze je k dispozici na *http://www.alvestrand.no/objectid/top.html* .

2) Zapojte pracoviště následovně:

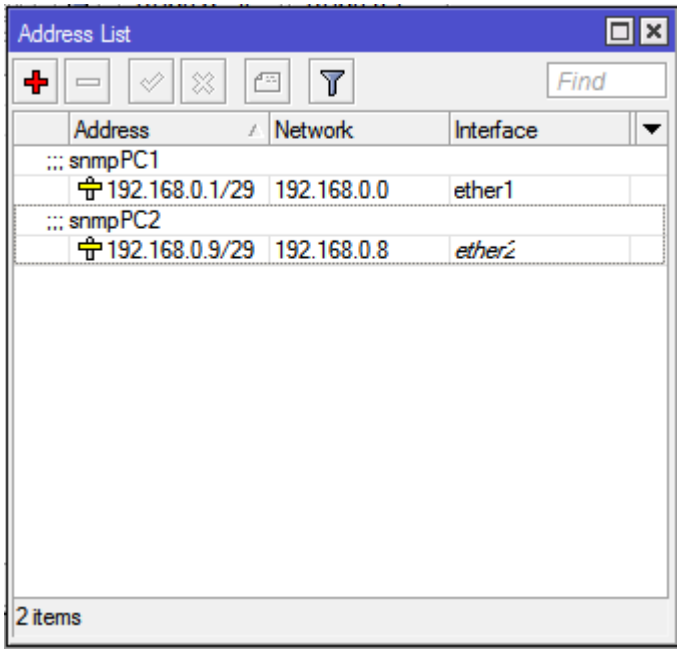


3) Nakonfigurujte směrovač. Použijte nastavení podsítí, jak je znázorněno na obrázku, případně zvolte své rozsahy dle vašeho uvážení.

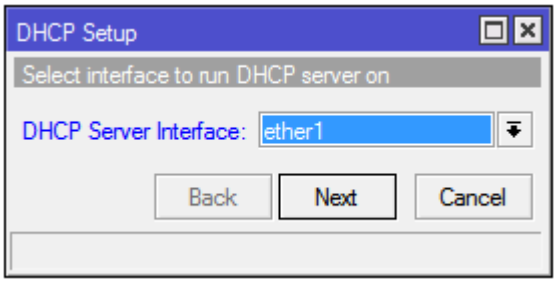
Konfigurace mikrotiku v bodech:

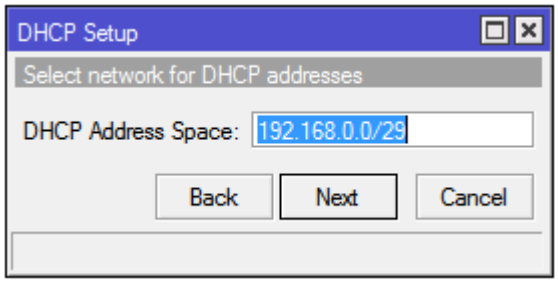
a) Otevřte program winbox a připojte se ke směrovači

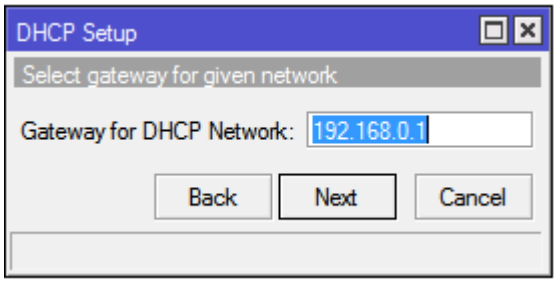
b) Vytvoření podsítí na rozhraních směrovače (IP->Addresses):

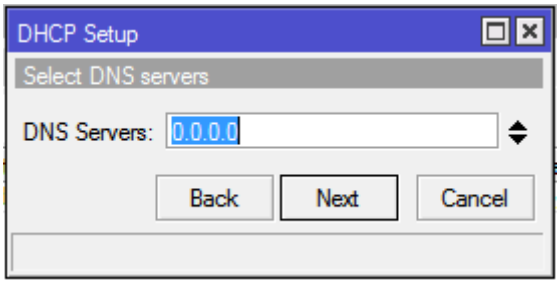


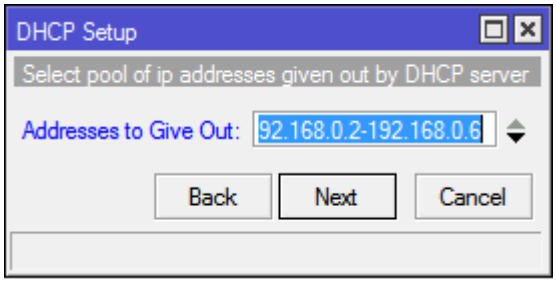
c) Zapnutí DHCP serveru pro obě rozhraní (IP->DHCP Server->DHCP Setup):



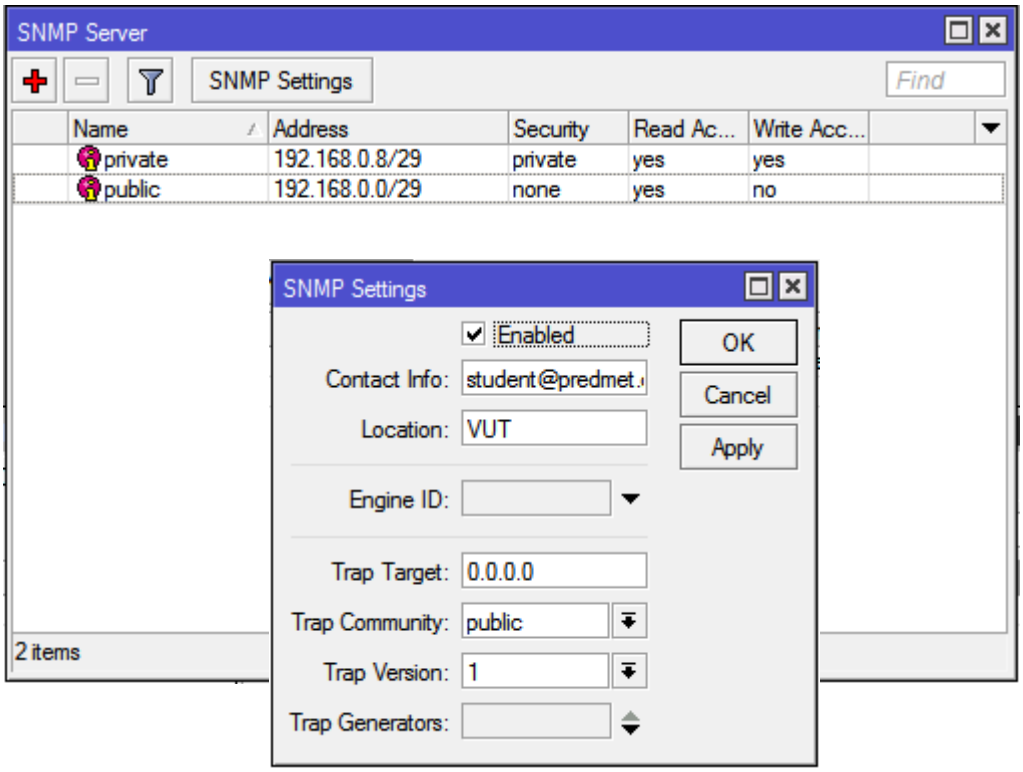








d) Nastavte SNMP Agenta (IP->SNMP):



4) Otevřete VMware Player na obou PC. Zkontrolujte nastavení sítě pro virtuální počítač SNMP_LAB_PC (Klikněte na *Edit virtual machine* a v záložce *Hardware* pod položkou *Network Adapter* zkontrolujte, zda je nastaveno *Network connection* na ***Bridged*** bez zatrhlé volby ***Replicate physical network connection state***. Poté spusťte oba virtuální počítače. Po spuštění Debianu se přihlašte pod uživatelem *root* a s heslem *test*.

5) Ověřte pomocí programu PING síťové propojení mezi virtuálními počítači.

6) Pomocí balíku NET-SNMP a operací SNMP zjištěte, o jaký typ směrovače se jedná, jaký je jeho *uptime*, jaký je zadán název směrovače a jak je pojmenováno jeho umístění. Některé z těchto parametrů již znáte, jelikož jste konfigurovali směrovač. Poznačte si však, jakými SNMP operacemi je zjistíte a které OID použijete. Dokážete tyto hodnoty zjistit pomocí jedné operace SNMP příkazu balíku NET-SNMP? Řešení si poznačte, bude kontrolováno vyučujícím.

7) Údaj uptime je udáván v tzv. TimeTicks. Přepočítejte tuto hodnotu na dny, hodiny a sekundy.

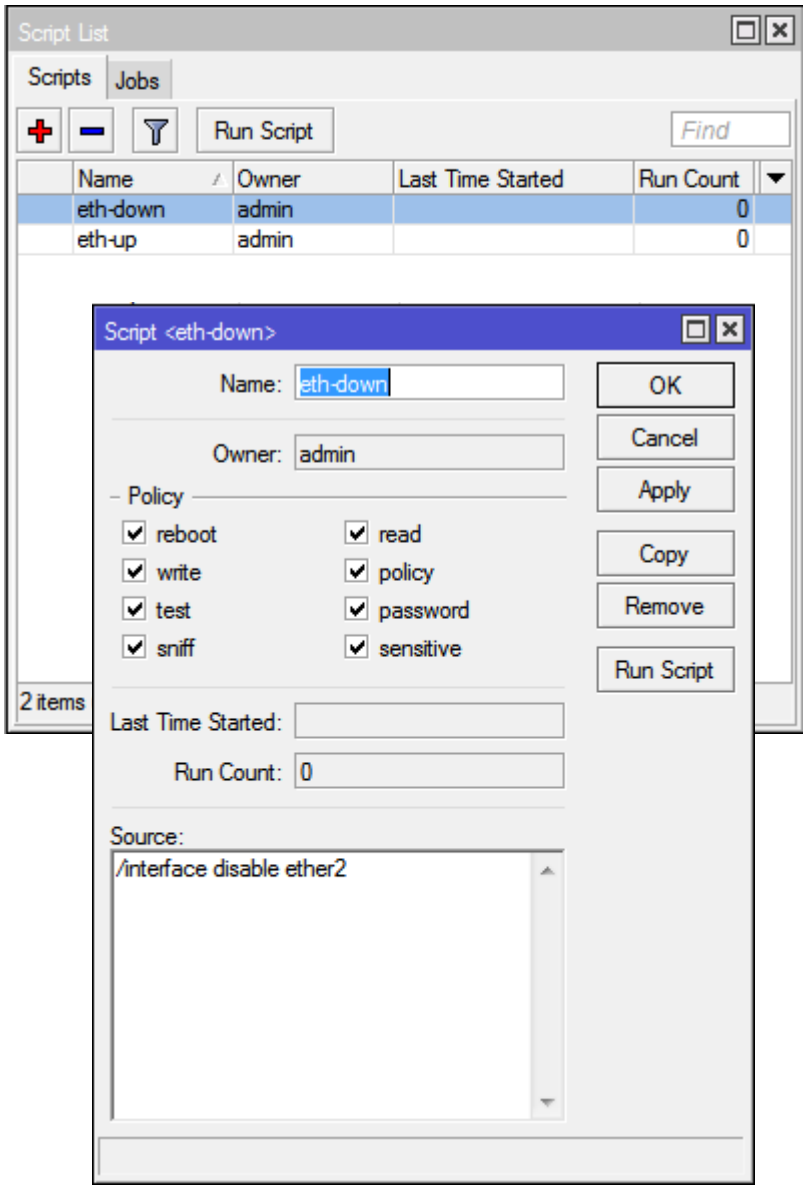
8) Pomocí operací SNMP a balíku NET-SNMP zjištěte objem přenesených dat přes rozhraní směrovače. Použijte strukturu MIB databáze pro nalezení správného čísla OID na *http://www.alvestrand.no/objectid/top.html*. Po zjištění konkretního OID si všimněte tytu jeho syntaxe. Vykonejte příkaz s tímto OID na rozhraní směrovače, na kterém budete sledovat provoz (*eth1* nebo *eth2*). Poznačte si navrácenou hodnotu. Poté přeneste soubor *data.lg* z jednoho virtuálního počítače na druhý a opět proveďte stejný SNMP příkaz. Z navrácených hodnot vypočítejte přibližnou velikost souboru, který byl přenášen. SNMP příkazy s OID si poznačte a navíc také vysvětlete, jak jste vypočítali velikost souboru. Pravděpodobně jste dospěli k větší velikosti souboru, než soubor veskutečnosti má. Čím je to způsobeno?

Pozn. 1: Soubor data.lg je umístěn v domovském adresáři uživatele user.

Pozn. 2: Přenesení dat lze realizovat pomocí programu SCP (Secure CoPy), který je součastí SSH (Secure SHell). Příklad použití SCP:

```
scp soubor_k_odeslani uzivateli@IPadresa:/cesta/kam/soubor/ulozit
```

9) Máte-li vše hotovo, vytvořte jednoduché skripty pro směrovač Mikrotik, které umožní vypnou a zapnout jeho rozhraní *eth2*. Tyto skripty vytvoříte v SYSTEM->SCRIPTS následovně:

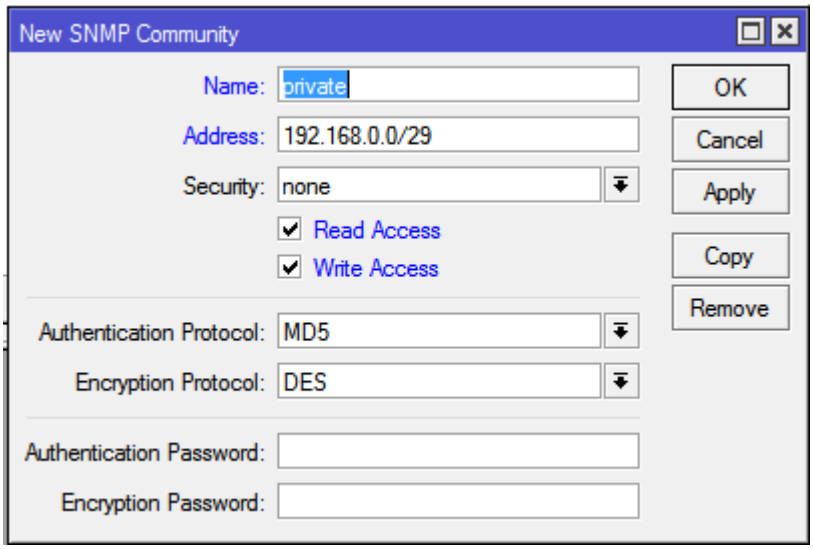


Rozhraní směrovače nelze vypnout přímo provedenim SNMP *set* dotazu na OID *ifAdmin Status*, jelikož to Mikrotik z bezpečnostních důvodu neumožňuje. Proto tuto možnost obcházíte vytvořením skriptu, který přes SNMP *set* operaci spustíme. SNMP *set* dotaz bude ve tvaru:

```
snmpset -c public -v 1 192.168.0.0 1.3.6.1.4.1.14988.1.1.8.1.1.3.X s 1
```

kde za X dosaďte číslo vašeho skriptu.

Přikaz se vám nevykoná, jelikož musíte vytvořit ještě pravidlo, umožňující zápis pro SNMP. Toto pravidlo nazvěte *private* a vytvořte jej následovně (v IP->SNMP):



Po provedení *snmpset* příkazu, kde změníte *public* na *private*, byste neměli být schopni úspěšně vykonat ping na rozhraní druhého virtuálního počítače nebo rozhraní *eth2* směrovače Mikrotik.

Doporučená literatura a odkazy

1) GARGULÁK, L. *Sběr dat o síťové komunikaci ze zařízení síťové infrastruktury*: Diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 94 stran. Vedoucí práce Ing. Radko Krkoš.

2) ALVESTRAND, H. Tveit: Object Identifiers [online].
Dostupné na <http://www.alvestrand.no/objectid>.

3) MIKROTIK: MikroTik RouterOS Scripting Manual [online].
Dostupné na <http://www.mikrotik.com/documentation/manual_2.5/Basic/Scripting.html>.

4) MIKROTIK: Manual:SNMP [online].
Dostupné na <http://wiki.mikrotik.com/wiki/SNMP>.

B Obsah přiloženého DVD disku

Samotná aplikace SDSKSI je uložena na disku DVD spolu s laboratorní úlohou a dalšími soubory. Před procházením DVD je doporučeno otevřít soubor *CtiMe.txt* umístěný v kořenovém adresáři. Disk DVD obsahuje tyto adresáře se soubory:

/LaboratorníÚloha

- OS Debian nakonfigurovaný pro lab. úlohu a spustitelný programem VMware Player
- Laboratorní úloha ve formátu pdf
- Laboratorní úloha v editovatelném formátu rtf

/Práce

- Diplomová práce ve formátu pdf
- OS Debian nakonfigurovaný s aplikaci SDSKSI a spustitelný programem VMware Player